Quality-adaptive Prefetching for Interactive Branched Video using HTTP-based Adaptive Streaming

Vengatanathan Krishnamoorthi[1], **Niklas Carlsson**[1],
Derek Eager[2], Anirban Mahanti[3], Nahid Shahmehri[1]

[1] Linköping university, Sweden
[2] University of Saskatchewan, Canada
[3] NICTA, Australia

Proc. ACM Multimedia, Orlando, FL, USA, Nov. 2014

We have all seen a movie that (in our taste) is...

We have all seen a movie that (in our taste) is...

too sad

We have all seen a movie that (in our taste) is...

too sad

too violent

We have all seen a movie that (in our taste) is...

too sad

too violent

too scary

...

We have all seen a movie that (in our taste) is...

too sad

too violent

too scary

...

… or where we may have wanted our favorite character to make a different choice...

We have all seen a movie that (in our taste) is...

too sad

too violent

too scary

...

… or where we may have wanted our favorite character to make a different choice...

We have all seen a movie that (in our taste) is...

too sad

too violent

too scary

...

… or where we may have wanted our favorite character to make a different choice...

We have all seen a movie that (in our taste) is...

too sad

too violent

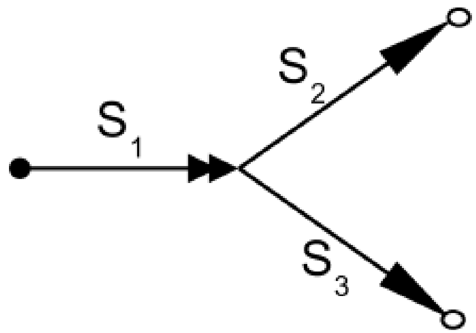
too scary

...

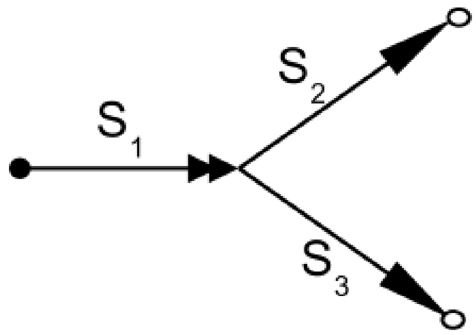
… or where we may have wanted our favorite character to make a different choice...

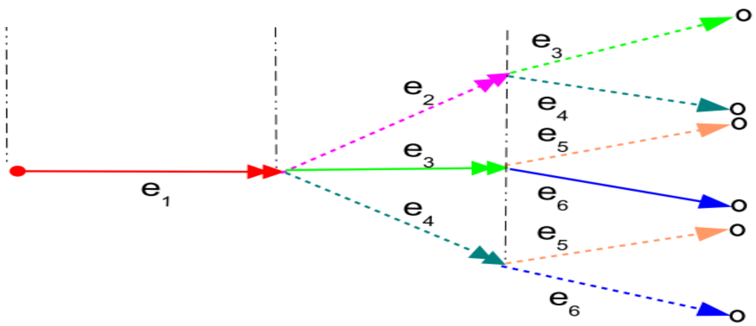
What if we can personalize the storyline based on the users preferences or path choices?

What if we can personalize the storyline based on the users preferences or path choices?

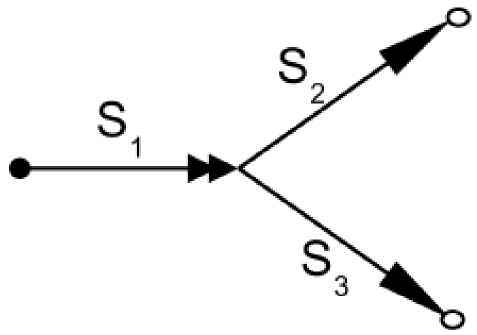


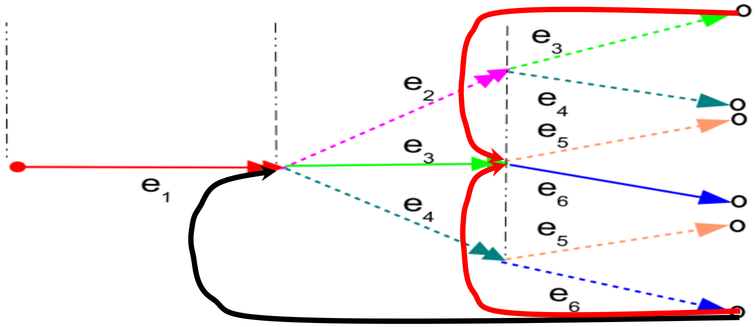
What if we can personalize the storyline based on the users preferences or path choices?



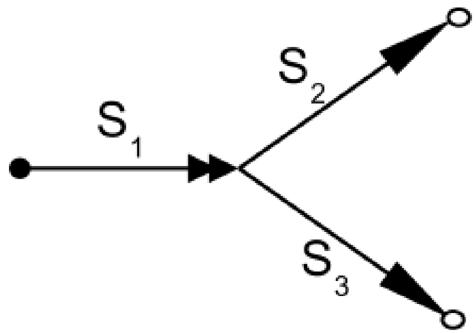


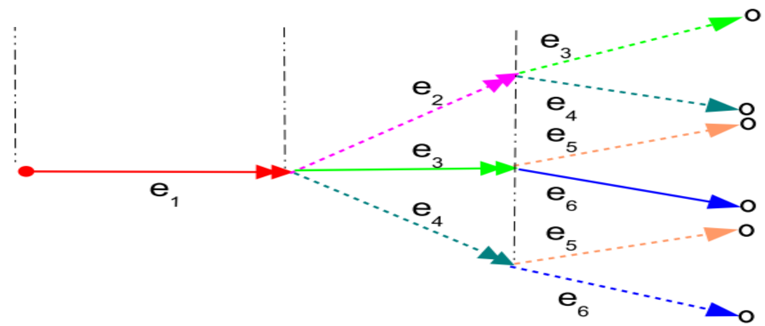
What if we can personalize the storyline based on the users preferences or path choices?





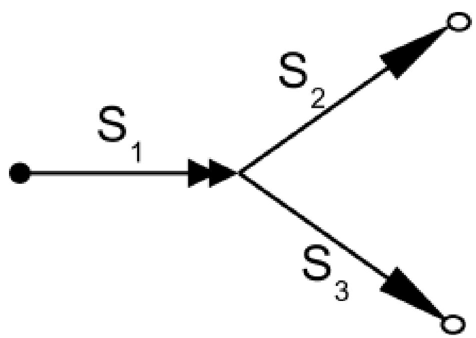
What if we can personalize the storyline based on the users preferences or path choices?

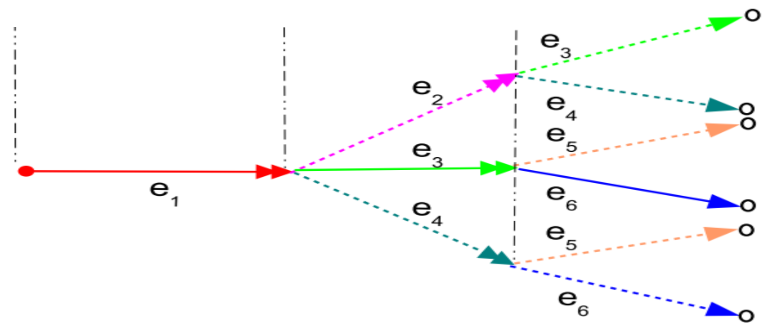




… already many examples how creative content creators provide interactive experiences and story lines …

What if we can personalize the storyline based on the users preferences or path choices?





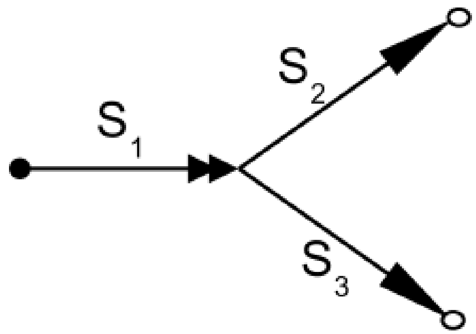
… already many examples how creative content creators provide interactive experiences and story lines …

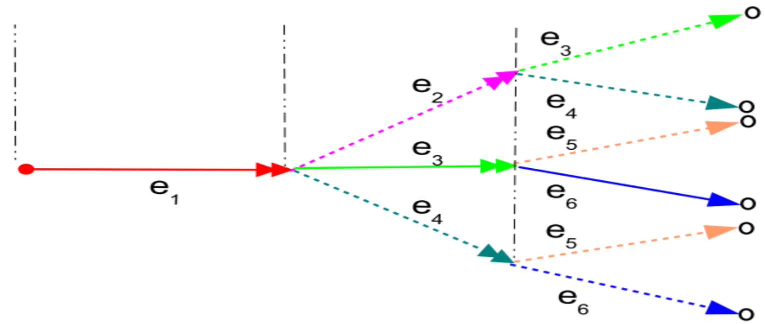
YouTube

Interlude

… and even books!

What if we can personalize the storyline based on the users preferences or path choices?





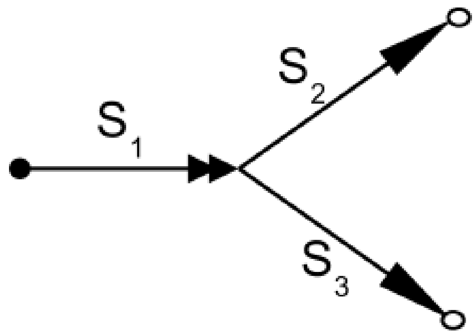
… already many examples how creative content creators provide interactive experiences and story lines …

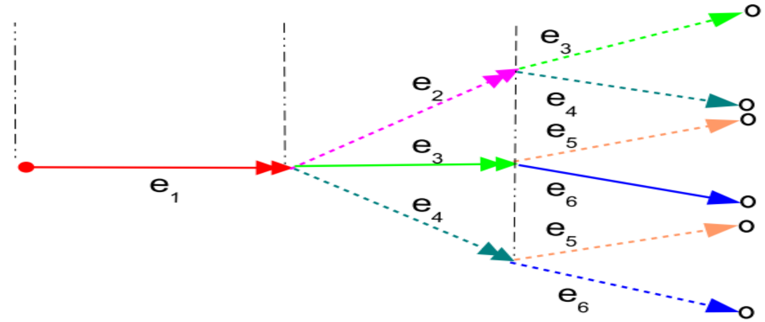
YouTube

Interlude

… and even books!

What if we can personalize the storyline based on the users preferences or path choices?

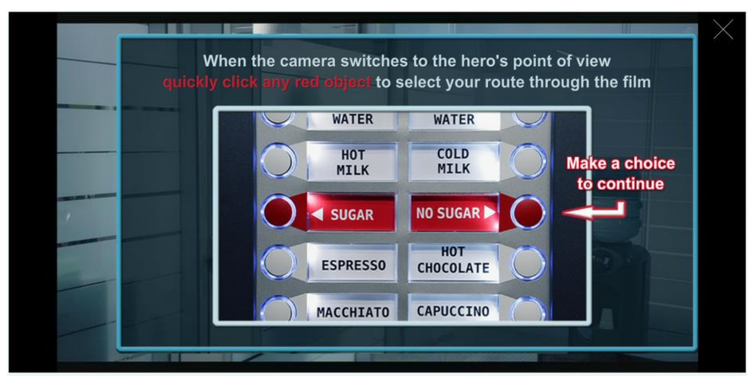




… already many examples how creative content creators provide interactive experiences and story lines …

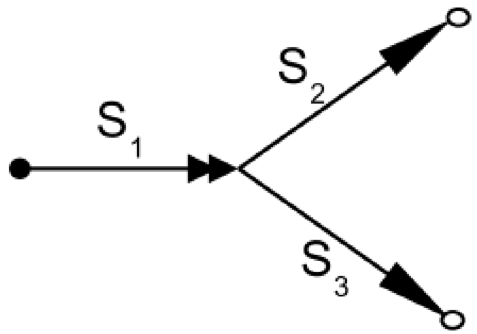
YouTube

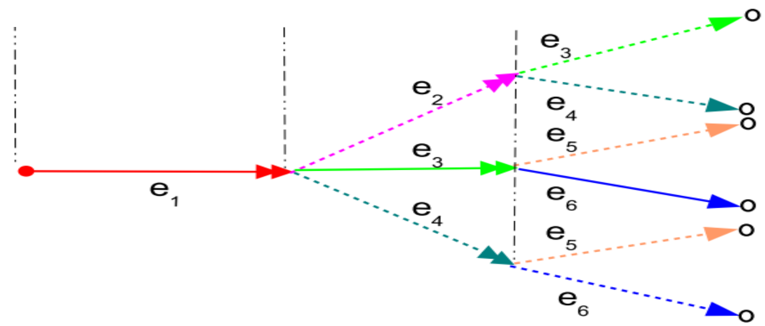
Interlude



… and even books!

What if we can personalize the storyline based on the users preferences or path choices?

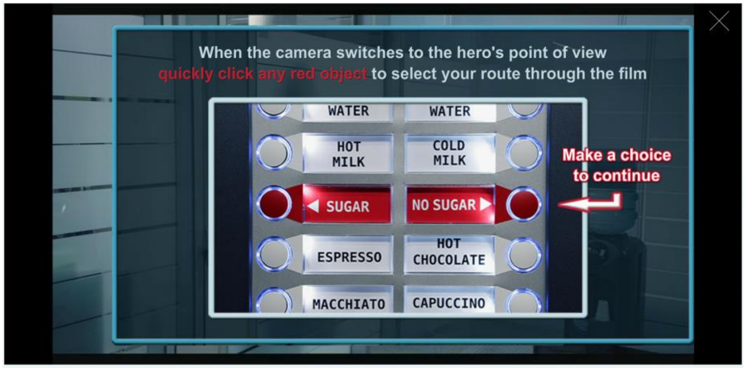




… already many examples how creative content creators provide interactive experiences and story lines …

YouTube

Interlude



… and even books!

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations

Seamless Playback without Stalls

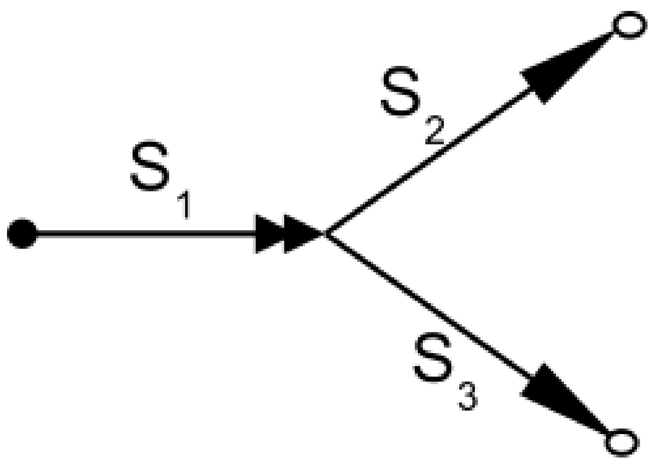
- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them



Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback
- Solution

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback
- Solution
  - Combine branched video and HAS

[Krishnamoorthi et al., *ACM CCR* 2013]

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback
- Solution
  - Combine branched video and HAS

[Krishnamoorthi et al., *ACM CCR* 2013]

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback
- Solution
  - Combine branched video and HAS

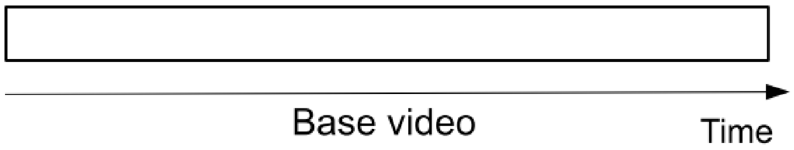
[Krishnamoorthi et al., *ACM CCR* 2013]

Seamless Playback without Stalls

- Regardless of interactivity, user experience and user satisfaction is greatly influenced by:
  - Playback stalls and quality fluctuations
- Current interactive branched players split a video into many sub videos and then link them
- Issues
  - Playback stalls when playing a new video
  - Non-adaptive playback
- Solution
  - Combine branched video and HAS

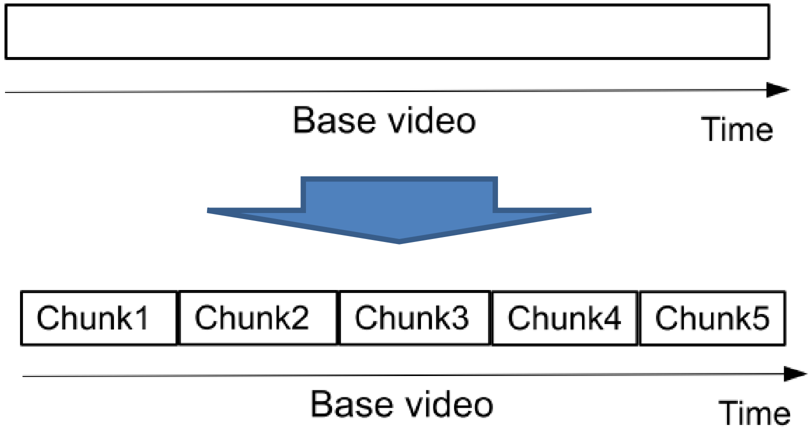
[Krishnamoorthi et al., *ACM CCR* 2013]

HTTP-based Adaptive Streaming (HAS)



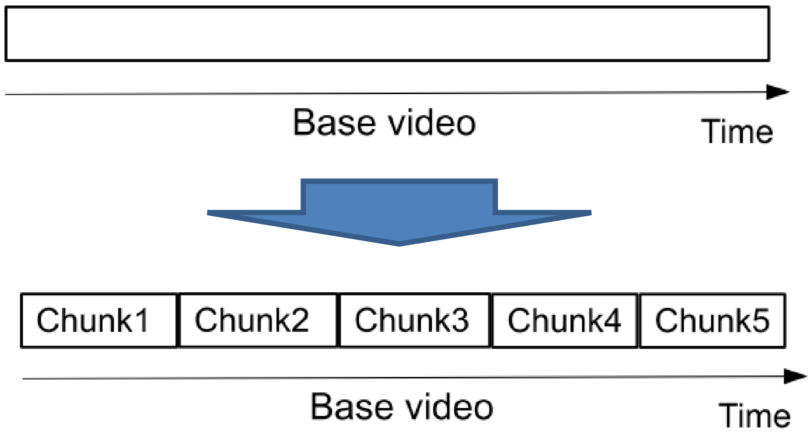
- HTTP-based streaming
  - Video is split into chunks

HTTP-based Adaptive Streaming (HAS)



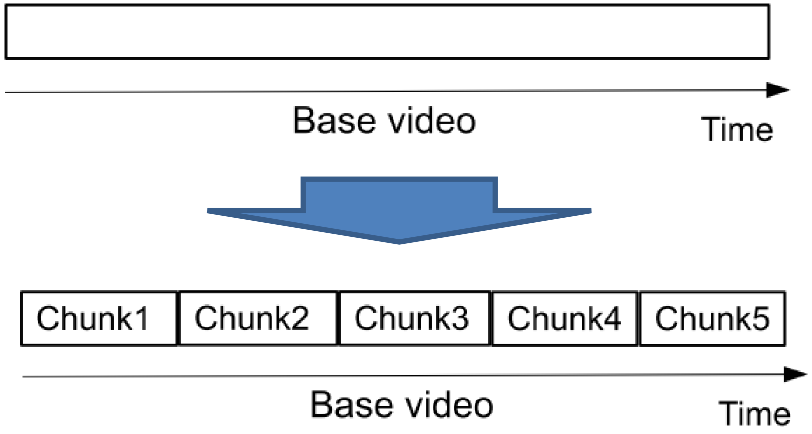
- HTTP-based streaming
  - Video is split into chunks

HTTP-based Adaptive Streaming (HAS)



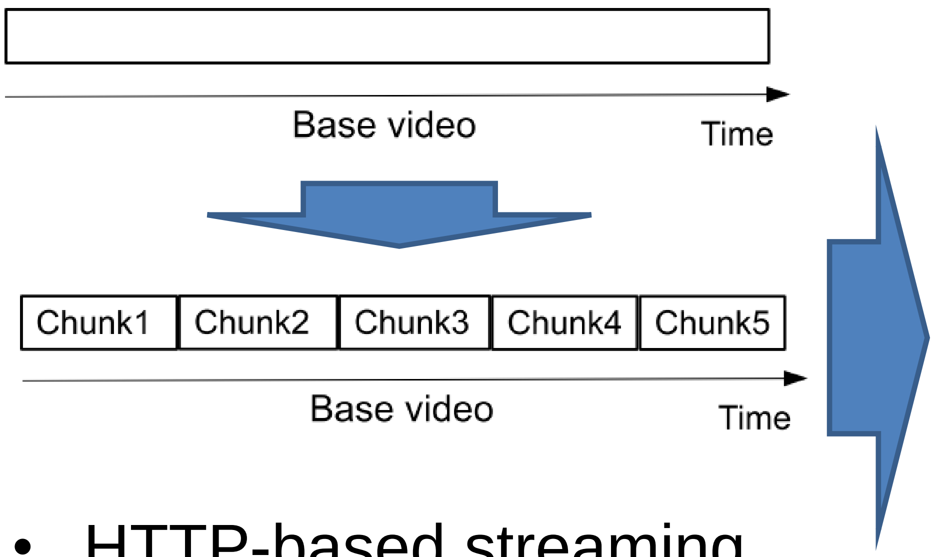
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching

HTTP-based Adaptive Streaming (HAS)



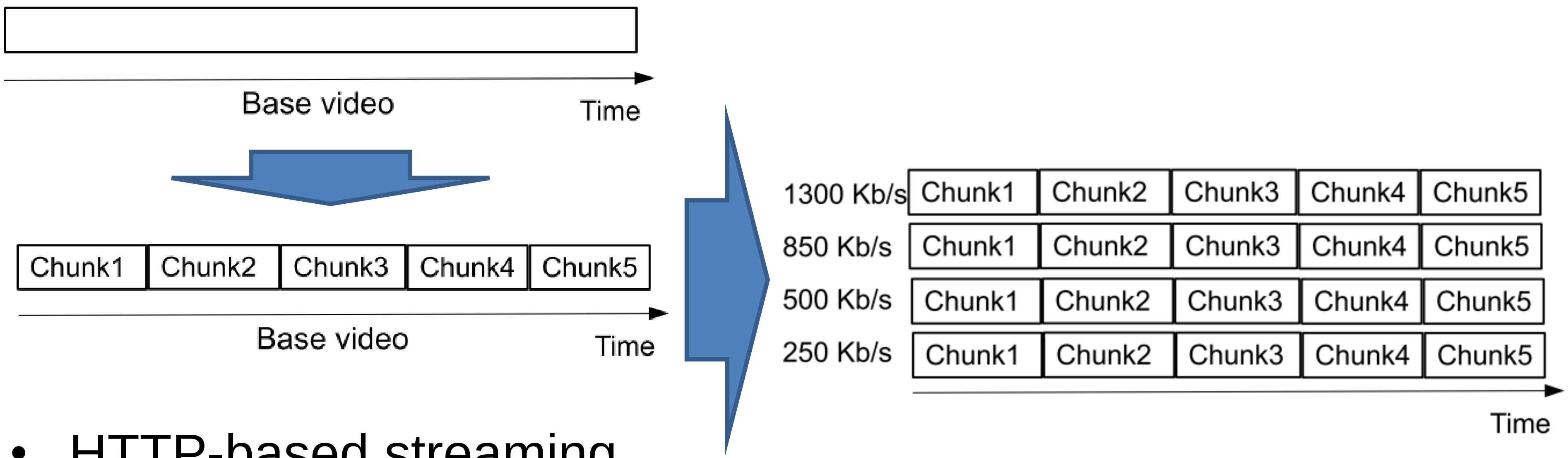
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD

HTTP-based Adaptive Streaming (HAS)



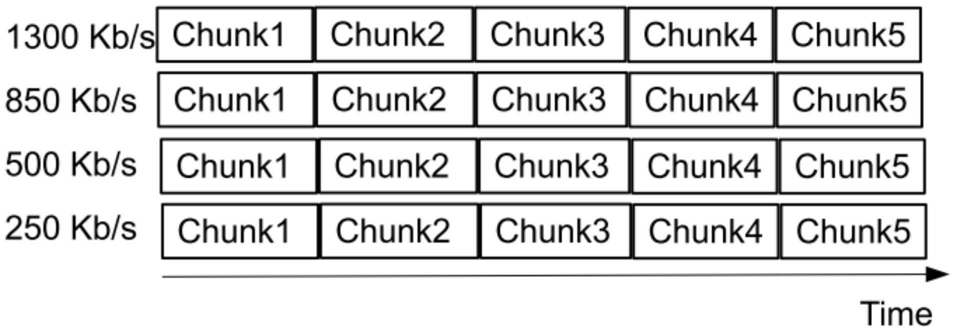
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming

HTTP-based Adaptive Streaming (HAS)



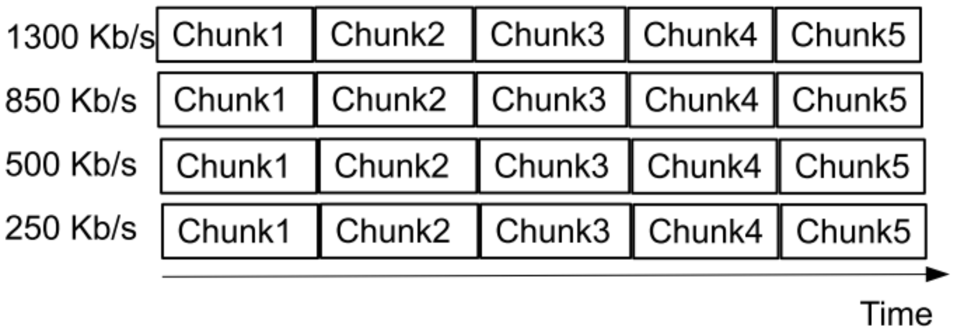
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)

HTTP-based Adaptive Streaming (HAS)



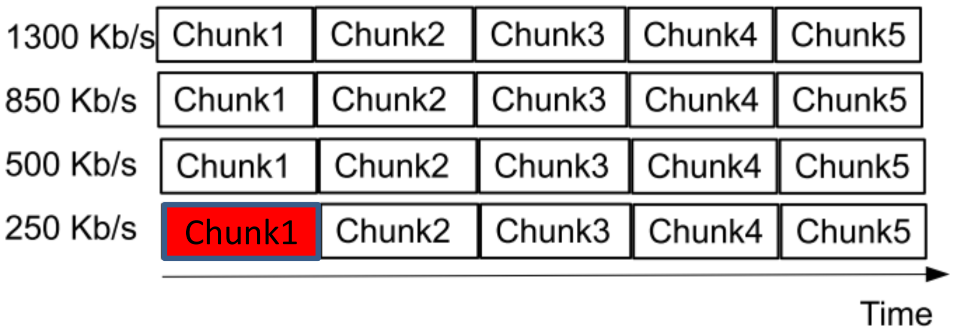
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)

HTTP-based Adaptive Streaming (HAS)



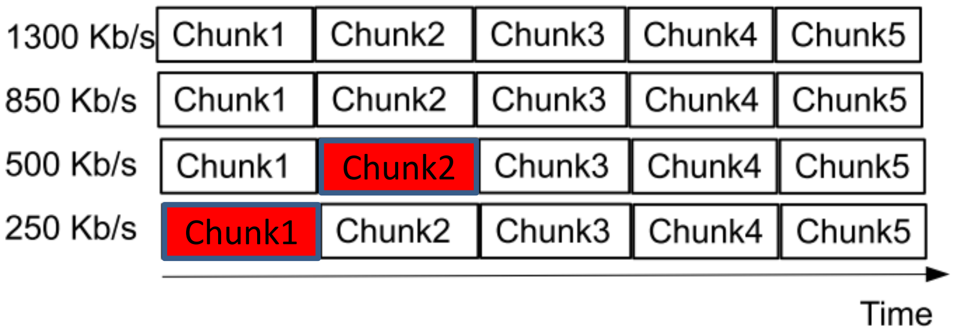
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



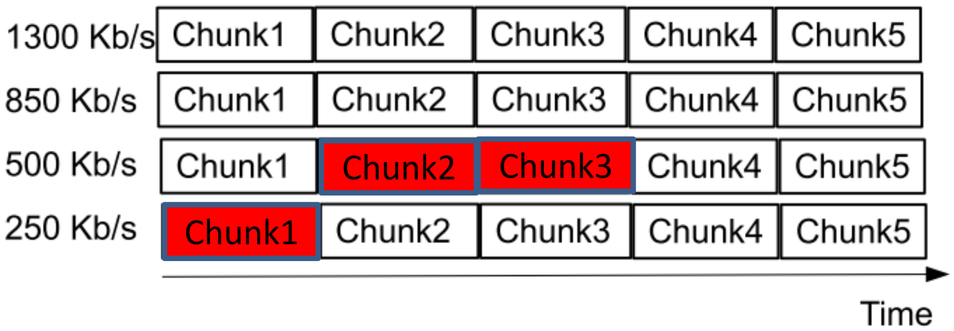
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



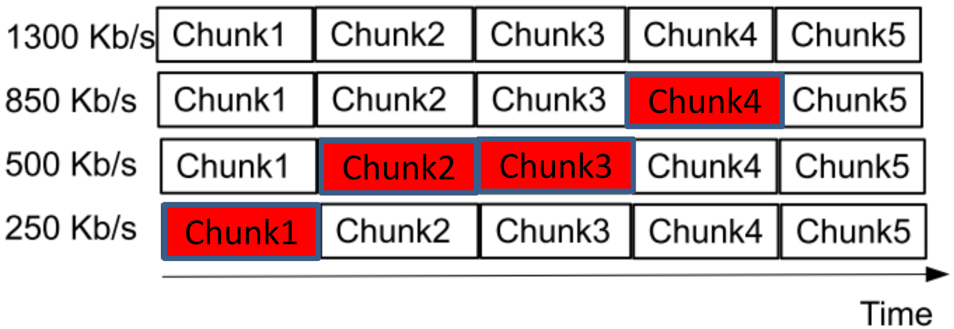
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



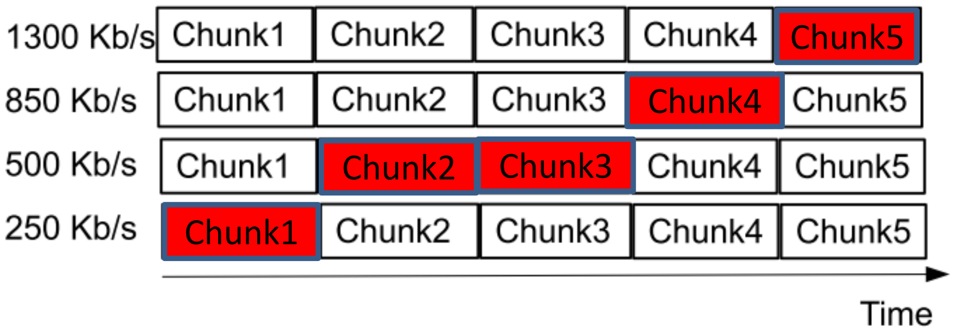
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



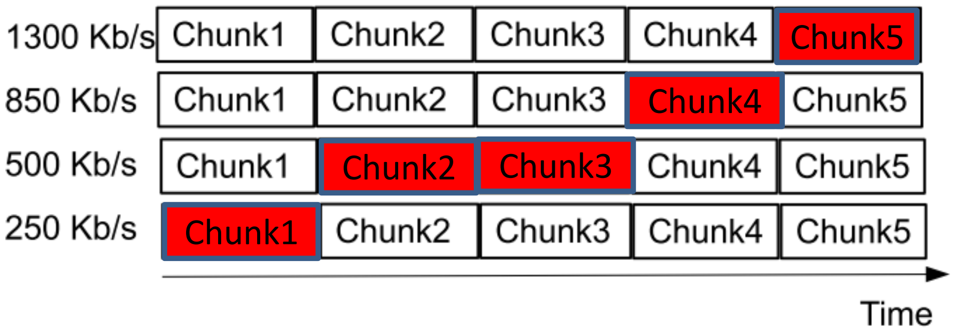
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



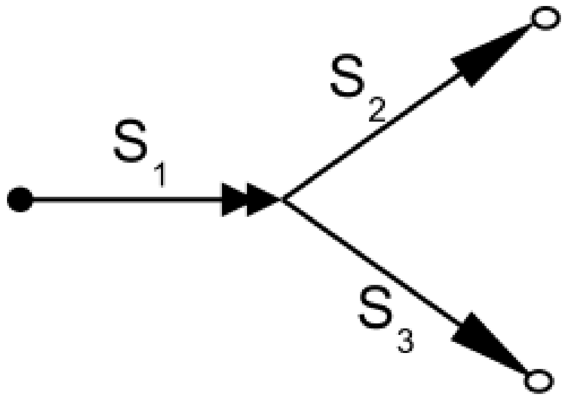
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HTTP-based Adaptive Streaming (HAS)



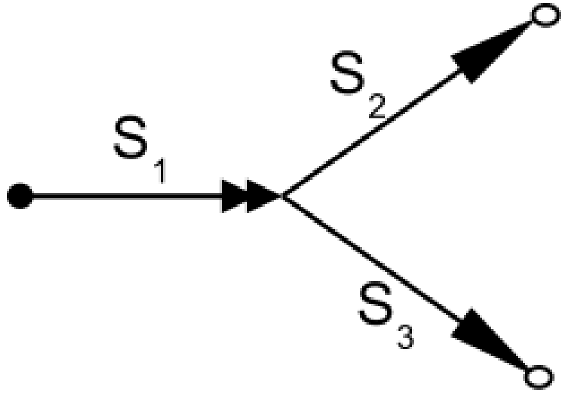
- HTTP-based streaming
  - Video is split into chunks
  - Easy firewall traversal and caching
  - Easy support for interactive VoD
- HTTP-based adaptive streaming
  - Multiple encodings of each fragment (defined in manifest file)
  - Clients adapt quality encoding based on buffer/network conditions

HAS-based Interactive Branched Video



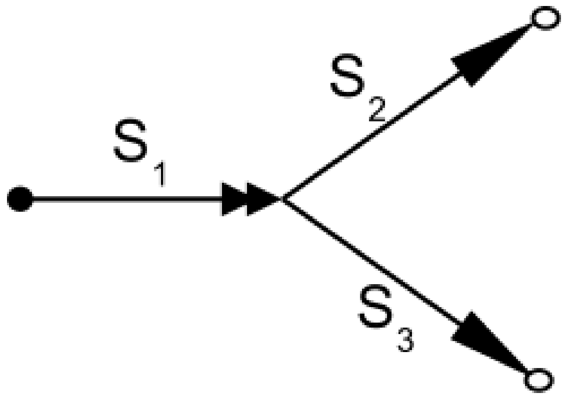
- Branched video and branch points

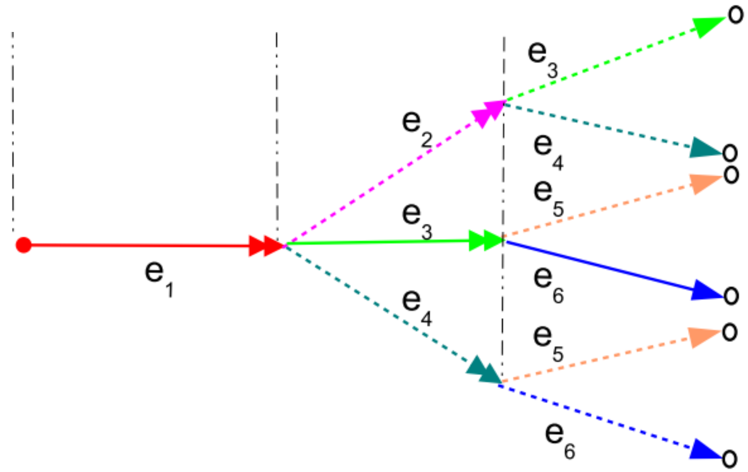
HAS-based Interactive Branched Video



- Branched video and branch points
  - The video can include branch points, with multiple branch choices

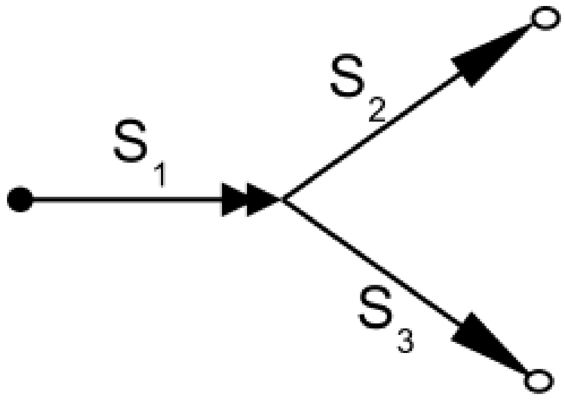
HAS-based Interactive Branched Video





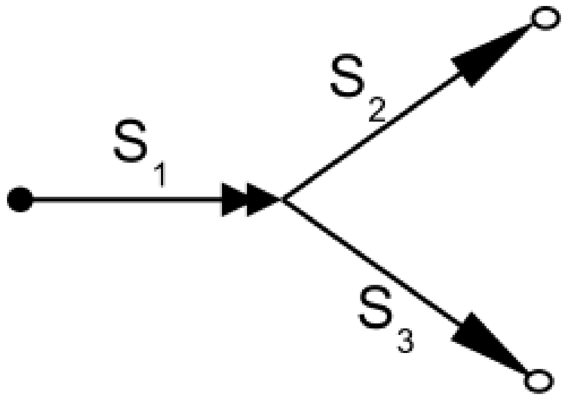
- Branched video and branch points
  - The video can include branch points, with multiple branch choices

HAS-based Interactive Branched Video



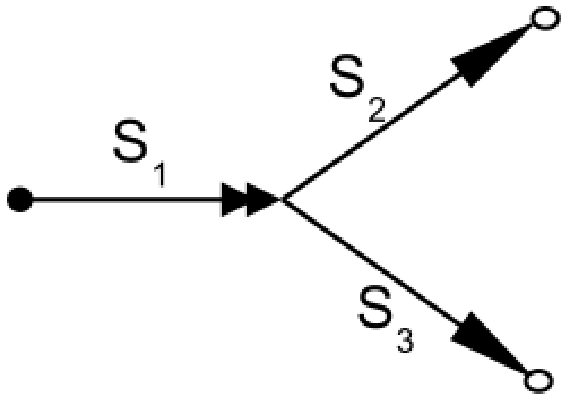
- Branched video and branch points
  - The video can include branch points, with multiple branch choices

HAS-based Interactive Branched Video



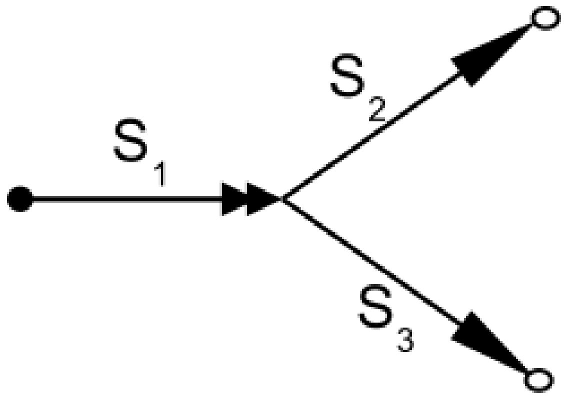
- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next

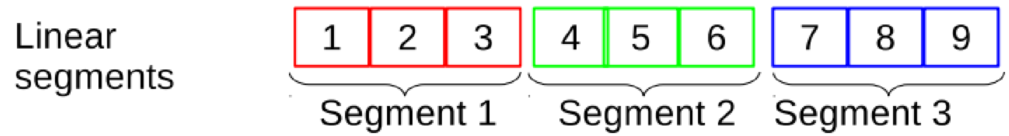
HAS-based Interactive Branched Video



- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments

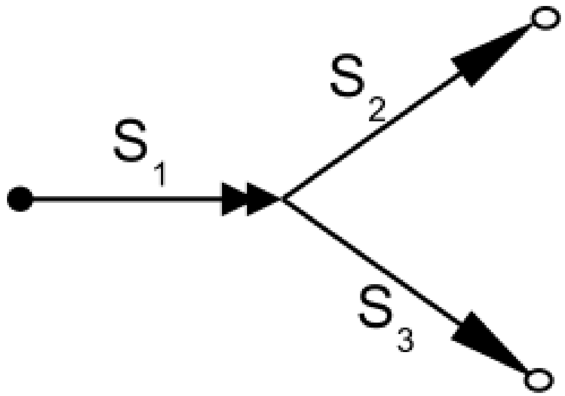
HAS-based Interactive Branched Video

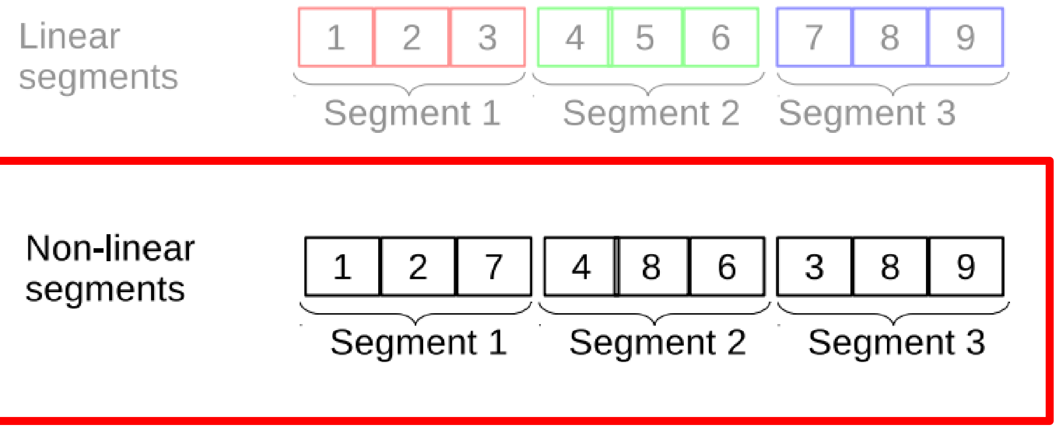




- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments
  - Arbitrary sequence of chunks from one or more videos

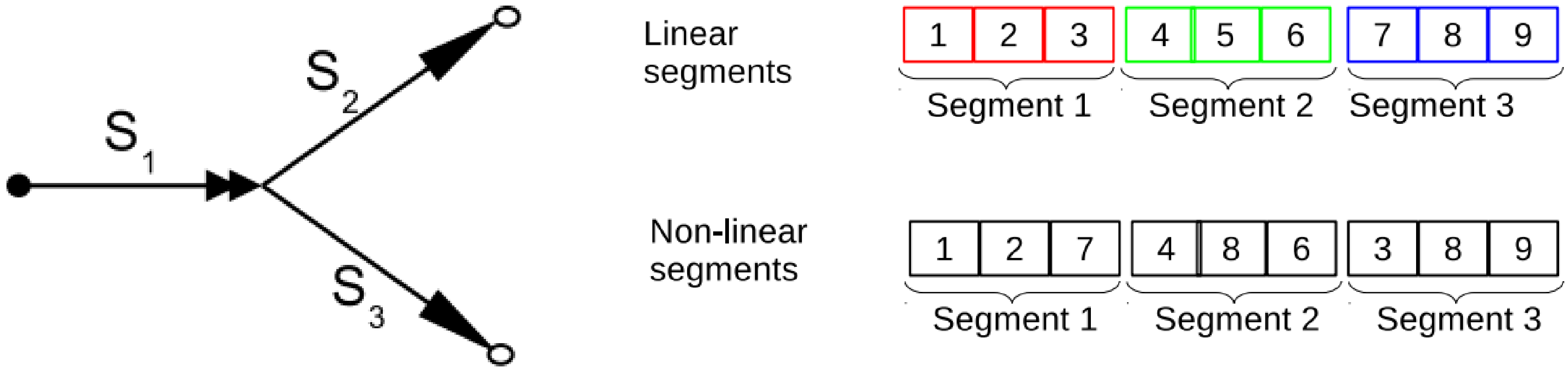
HAS-based Interactive Branched Video





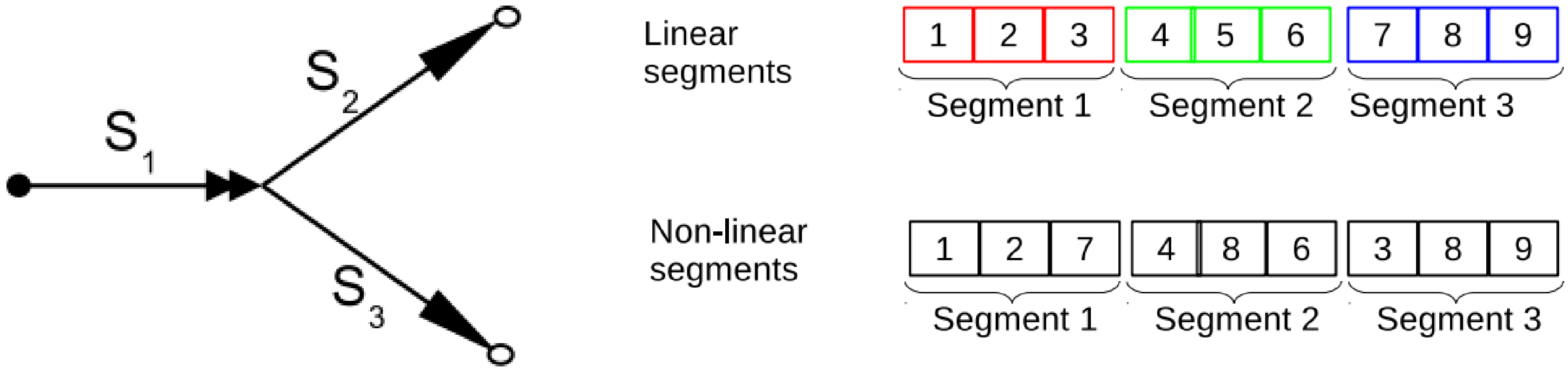
- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments
  - Arbitrary sequence of chunks from one or more videos

HAS-based Interactive Branched Video



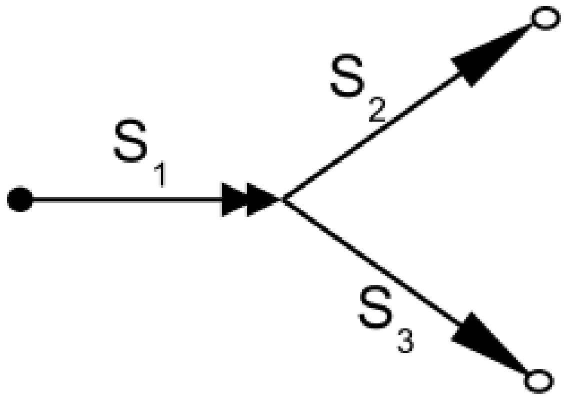
- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments
  - Arbitrary sequence of chunks from one or more videos
- Use of HAS allow adaptive prefetching

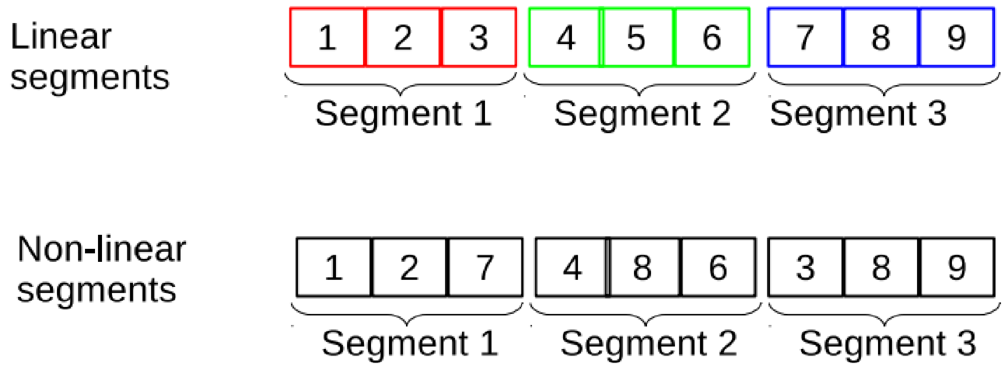
HAS-based Interactive Branched Video



- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments
  - Arbitrary sequence of chunks from one or more videos
- Use of HAS allow adaptive prefetching
  - Goal: Seamless playback even if user decision at last possible moment

HAS-based Interactive Branched Video





- Branched video and branch points
  - The video can include branch points, with multiple branch choices
  - User selects which segment to play back next
- Segments
  - Arbitrary sequence of chunks from one or more videos
- Use of HAS allow adaptive prefetching
  - Goal: Seamless playback even if user decision at last possible moment

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation* of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

**Software: http://www.ida.liu.se/~nikca/mm14.html*

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation* of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

**Software: http://www.ida.liu.se/~nikca/mm14.html*

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation* of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

**Software: http://www.ida.liu.se/~nikca/mm14.html*

Contributions

- We develop a simple analytic model which allows us to define the prefetching problem as an optimization problem
  - Maximizes expected playback quality while avoiding stalls
- Based on our findings, we design optimized policies that determine:
  1. When different chunks should be downloaded
  2. What quality level should be selected for each of these chunks
  3. How to manage playback buffers and (multiple) TCP connections

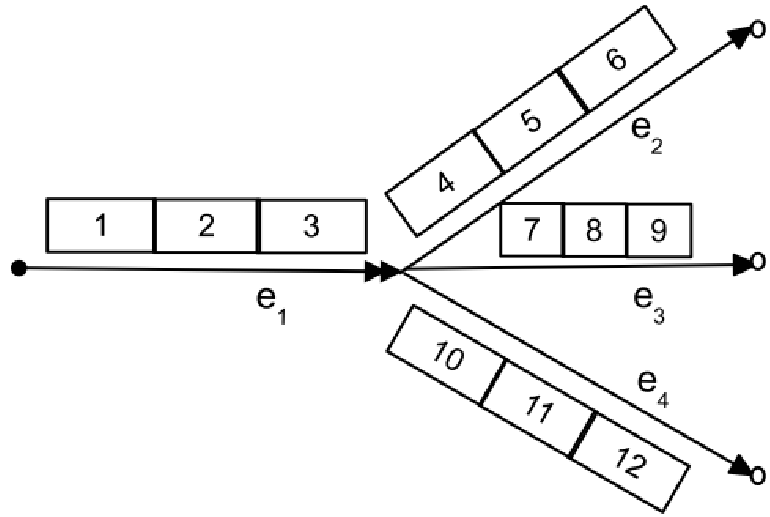
  such as to ensure smooth playback experience without excessive workahead (buffering)
- The design and implementation* of the framework
- Experimental evaluation of our policies, which provide insights into the importance of careful adaptive policies

**Software: http://www.ida.liu.se/~nikca/mm14.html*

Problem Description and Constraints

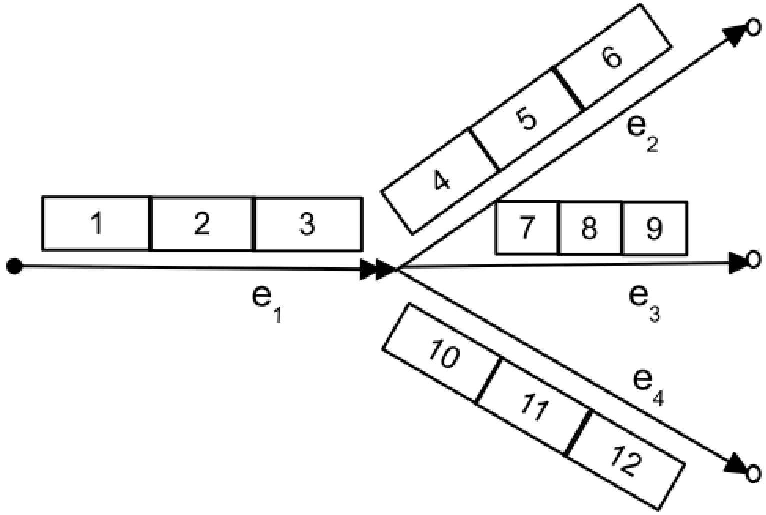
- Problem: Maximize quality, given playback deadlines and bandwidth conditions

Problem Description and Constraints



- Problem: Maximize quality, given playback deadlines and bandwidth conditions

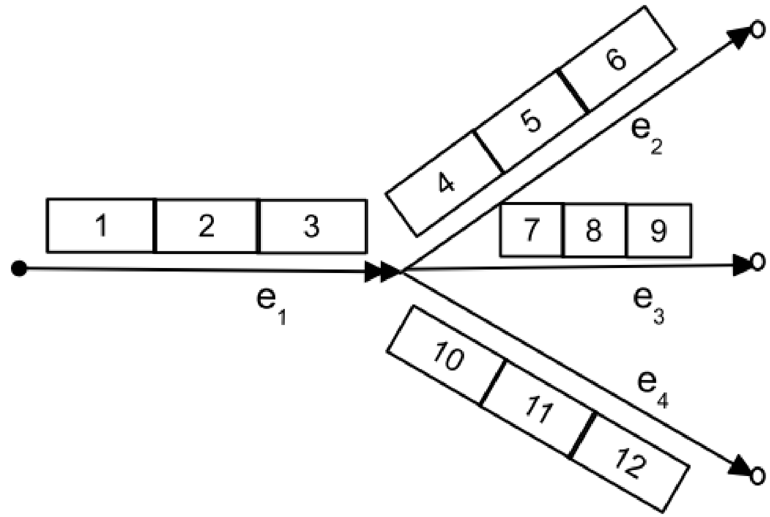
Problem Description and Constraints



- Objective function

maximize playback quality

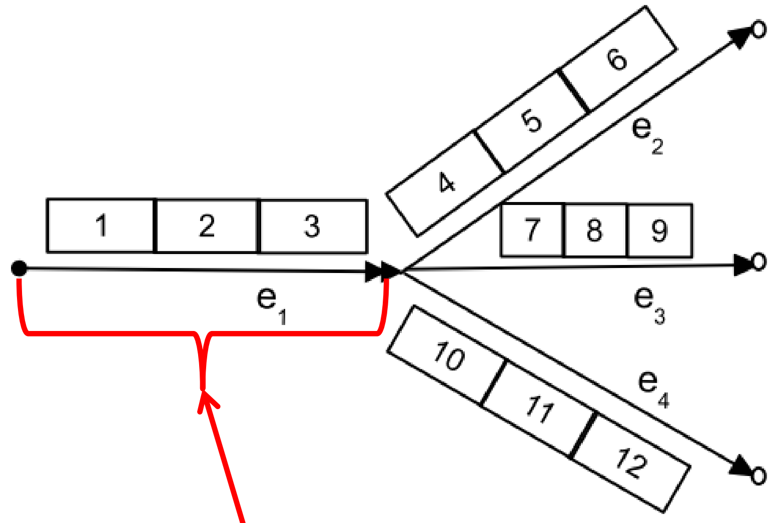
Problem Description and Constraints



- Objective function

$$\text{maximize} \sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+|\mathcal{E}^b|} q_i l_i$$

Problem Description and Constraints

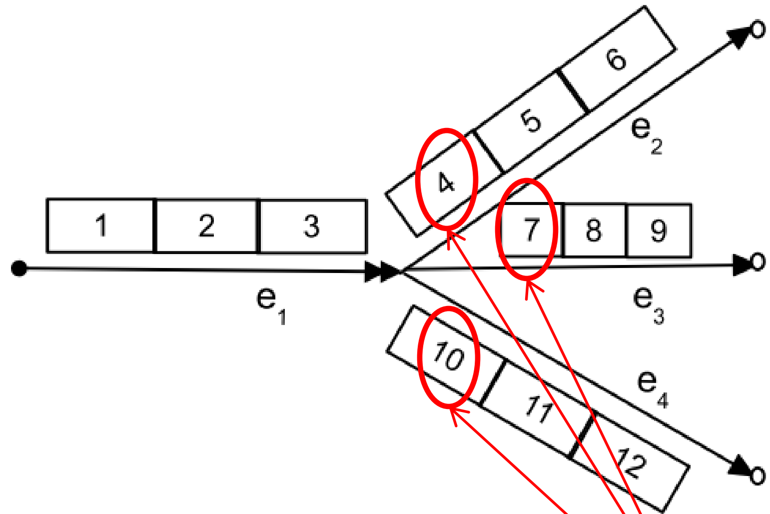


- Objective function

$$\text{maximize} \sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+|\mathcal{E}^b|} q_i l_i$$

Current segment

Problem Description and Constraints

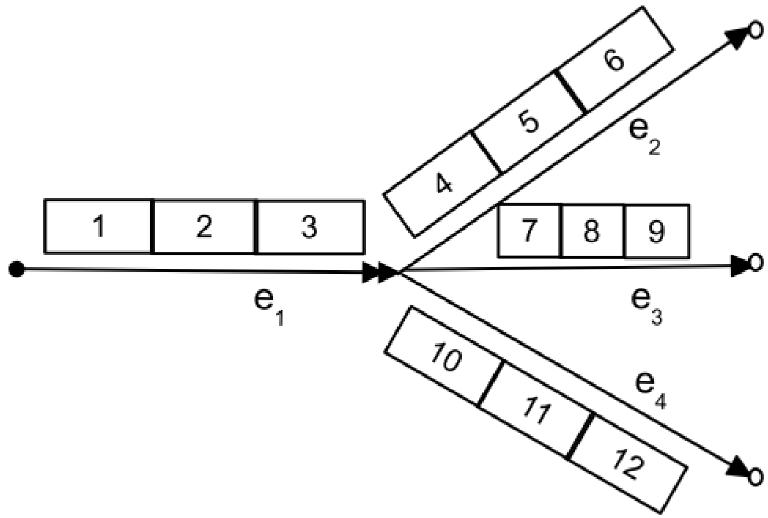


- Objective function

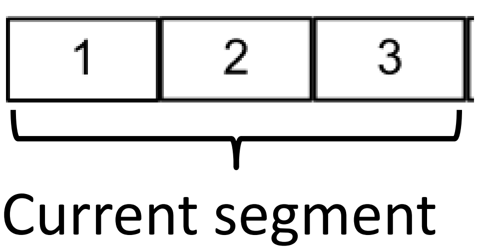
$$\text{maximize} \sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+|\mathcal{E}^b|} q_i l_i$$

Beginning of next segment

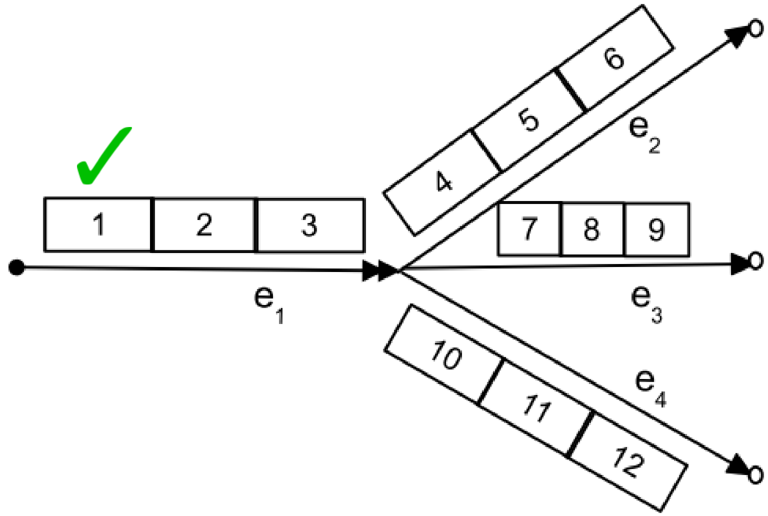
Problem Description and Constraints



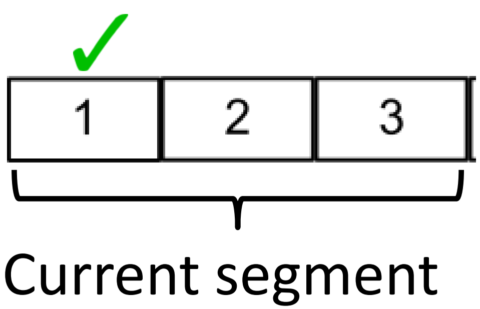
- Download order: round robin (optimal)



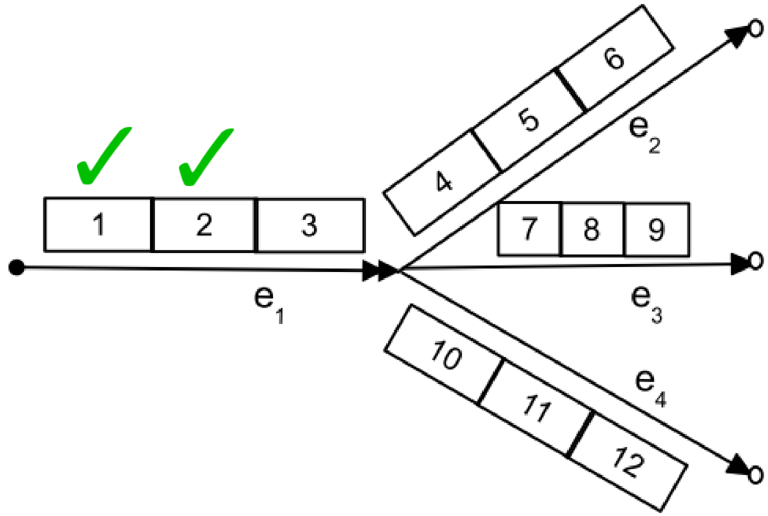
Problem Description and Constraints



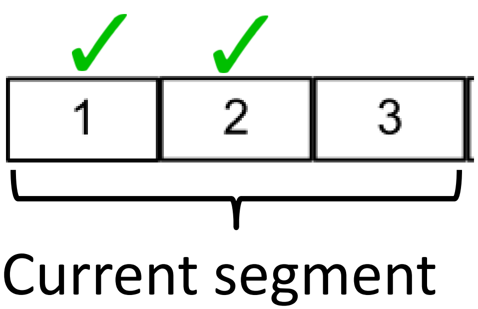
- Download order: round robin (optimal)



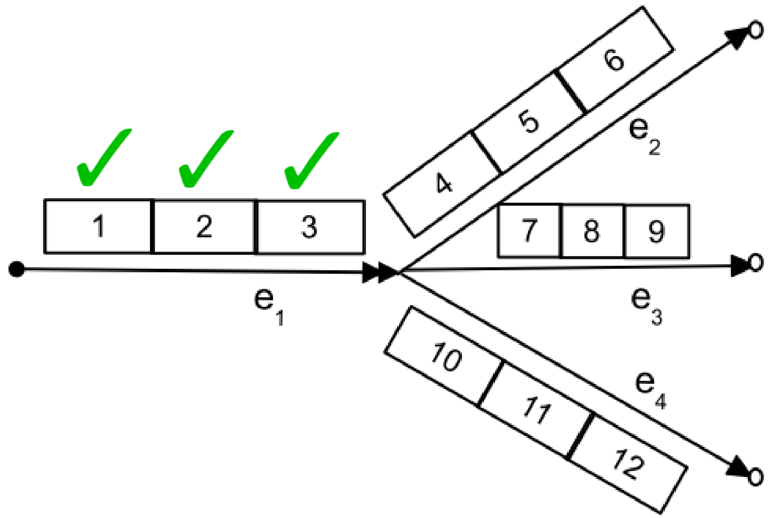
Problem Description and Constraints



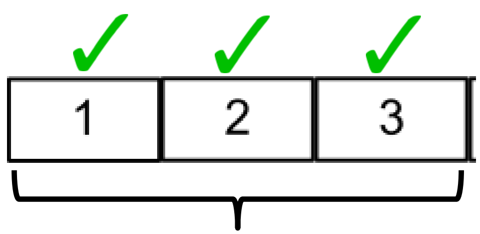
- Download order: round robin (optimal)



Problem Description and Constraints

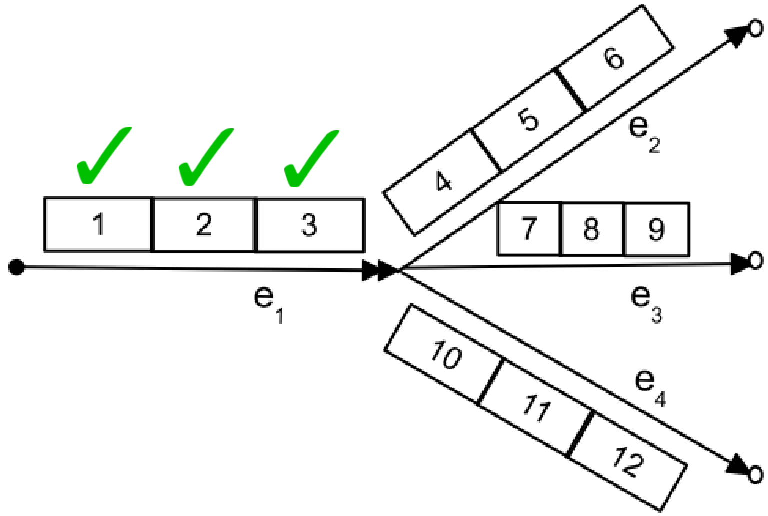


- Download order: round robin (optimal)

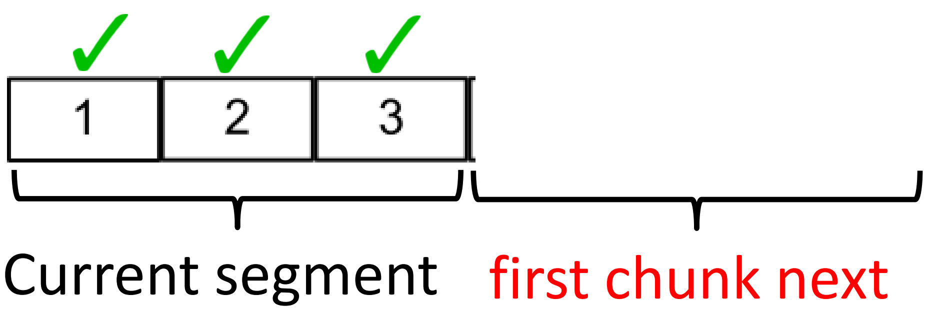


Current segment

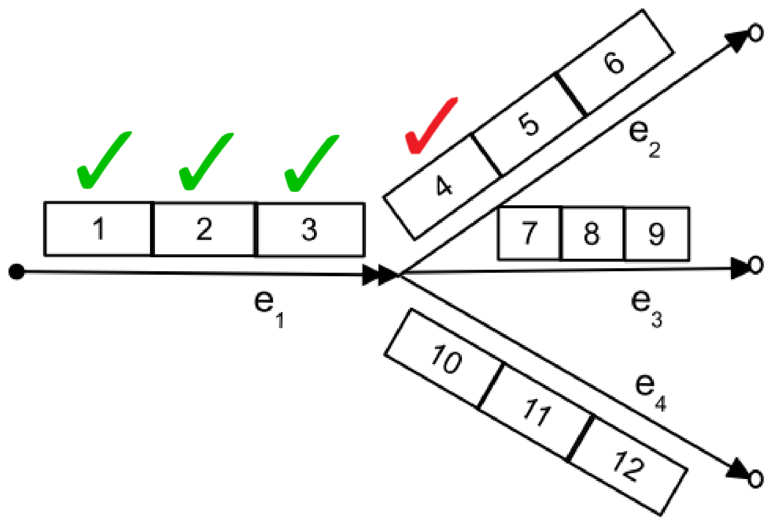
Problem Description and Constraints



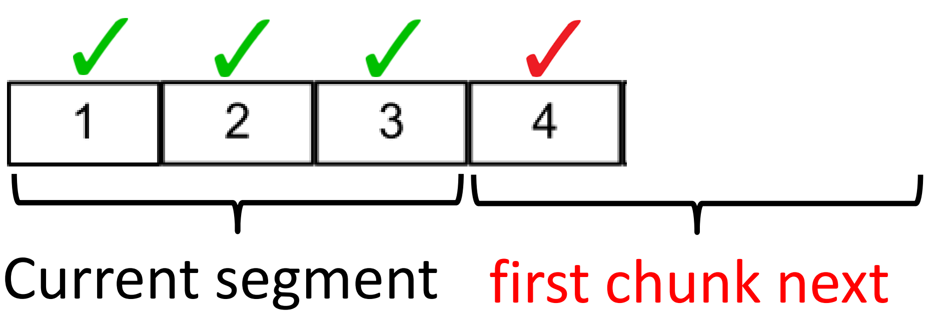
- Download order: round robin (optimal)



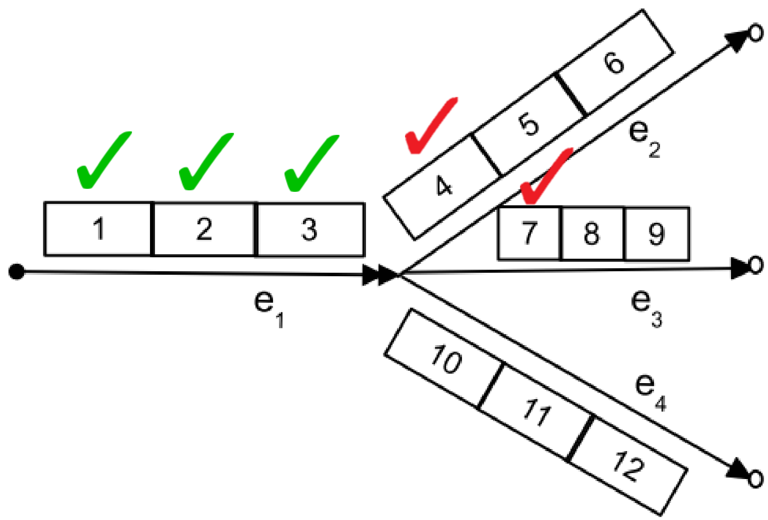
Problem Description and Constraints



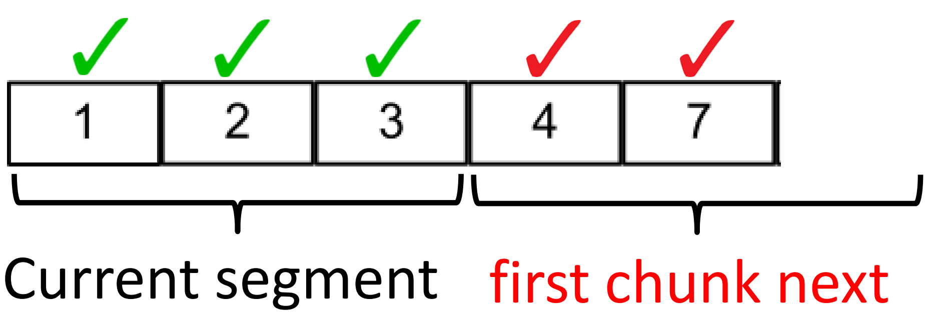
- Download order: round robin (optimal)



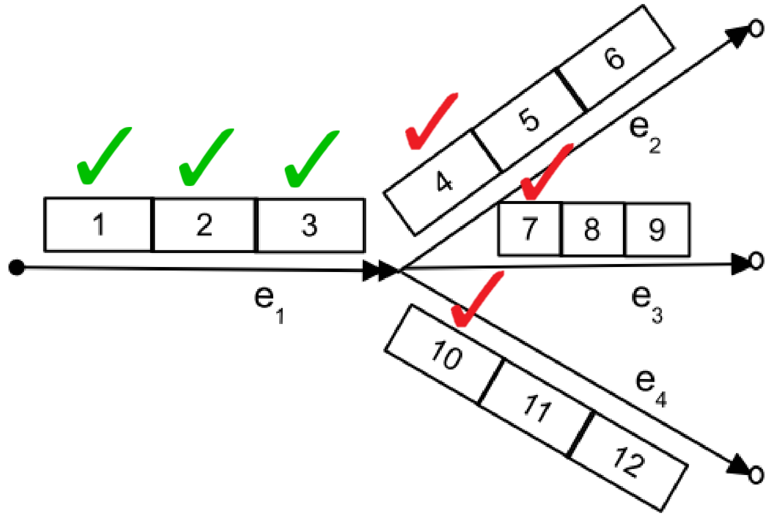
Problem Description and Constraints



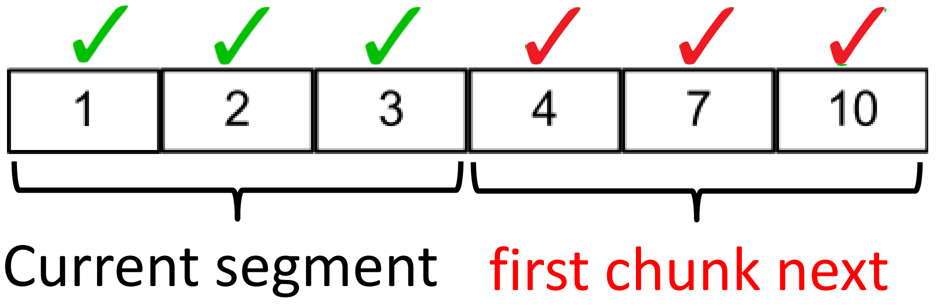
- Download order: round robin (optimal)



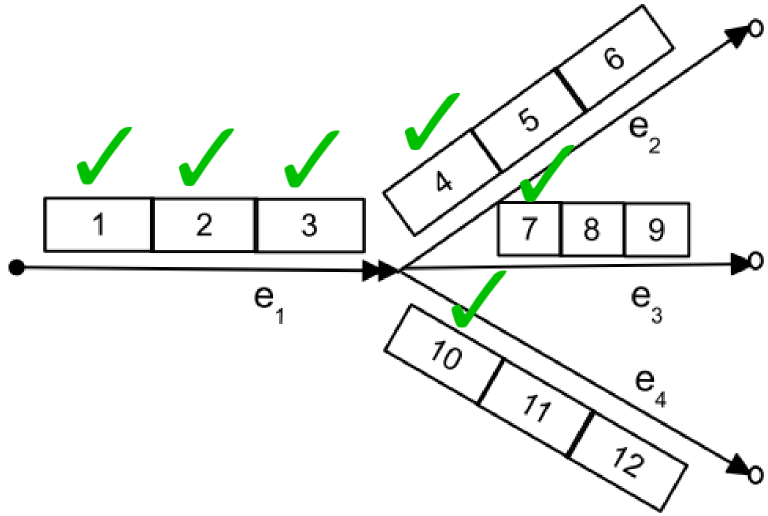
Problem Description and Constraints



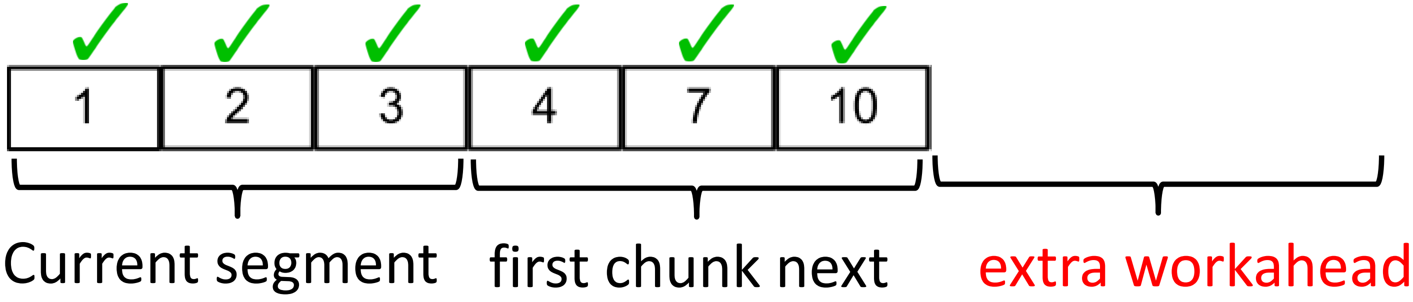
- Download order: round robin (optimal)



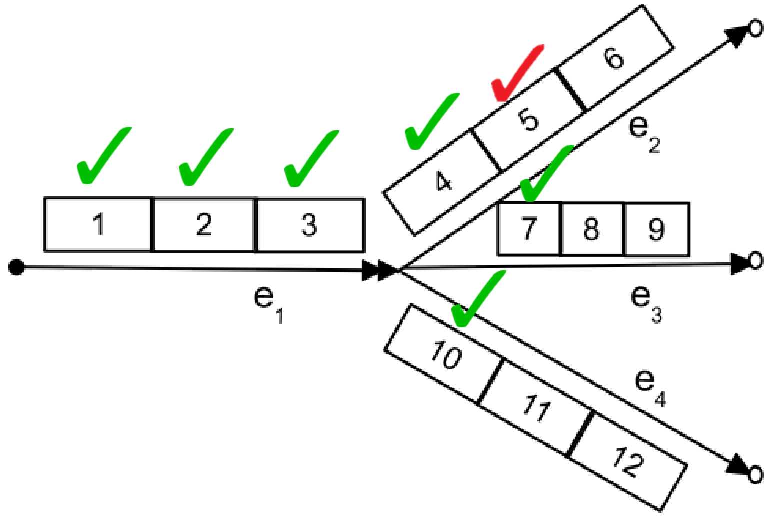
Problem Description and Constraints



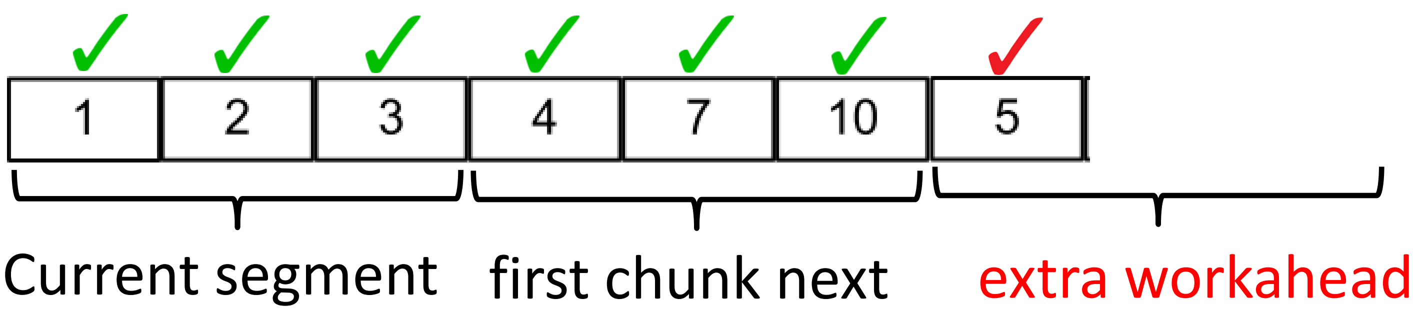
- Download order: round robin (extra workahead)



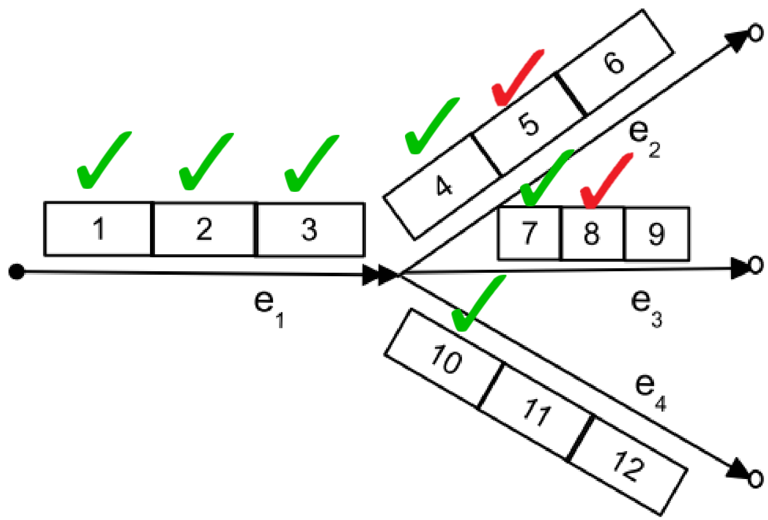
Problem Description and Constraints



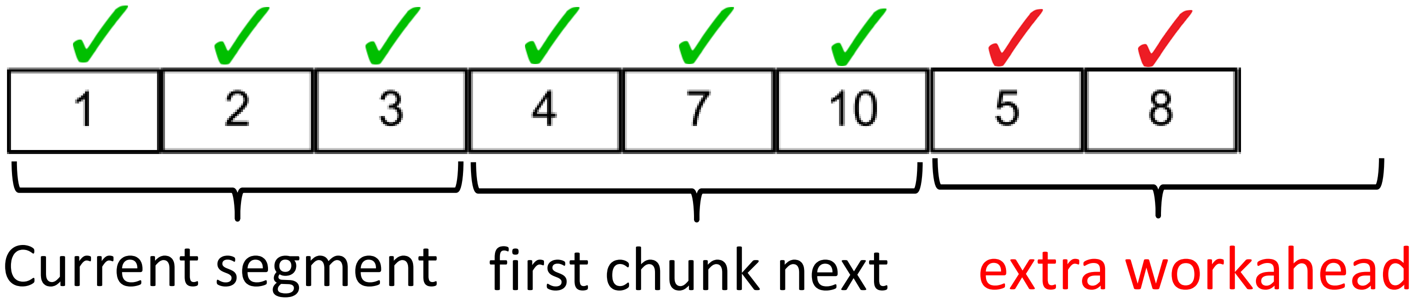
- Download order: round robin (extra workahead)



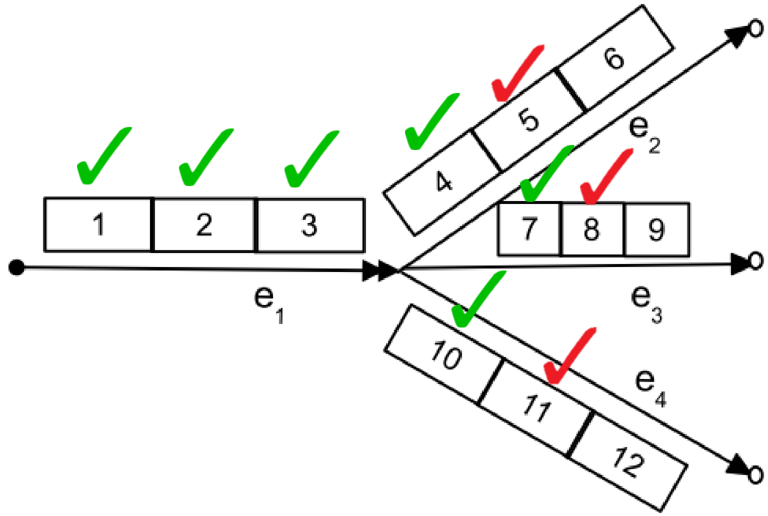
Problem Description and Constraints



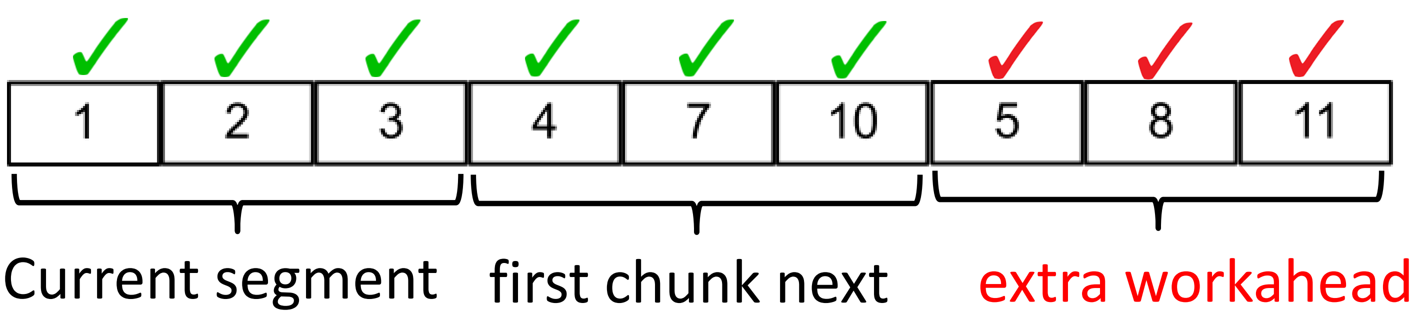
- Download order: round robin (extra workahead)



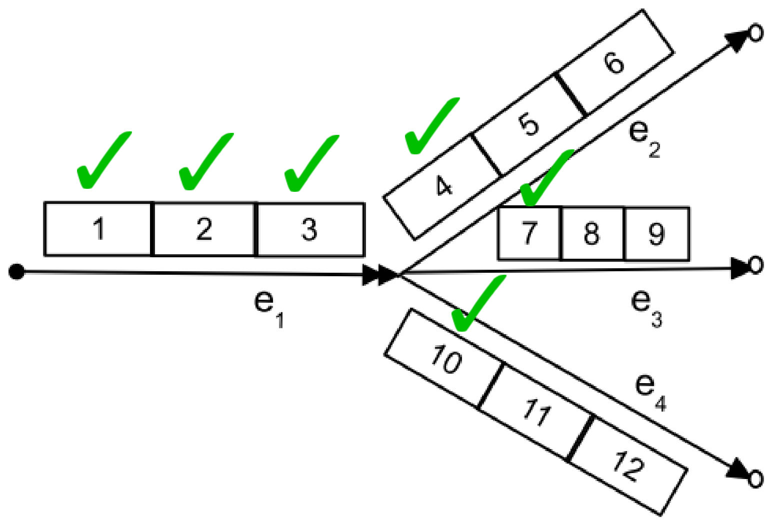
Problem Description and Constraints



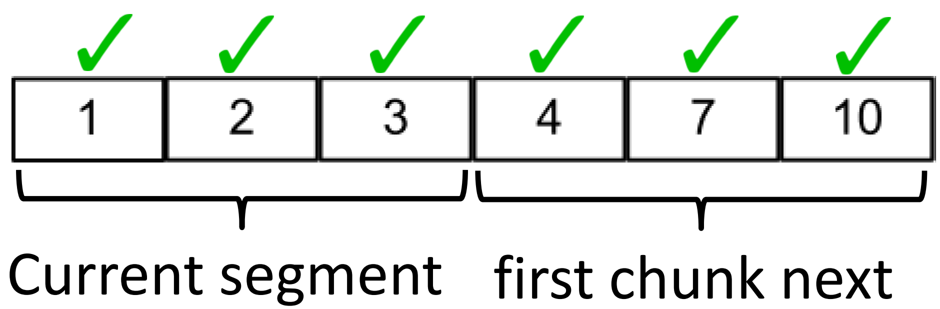
- Download order: round robin (extra workahead)



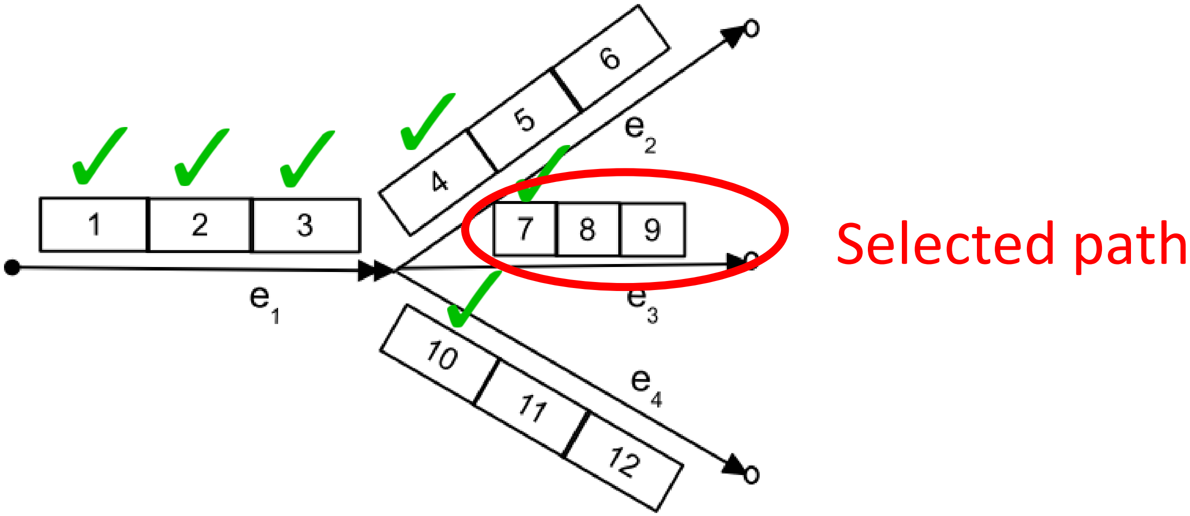
Problem Description and Constraints



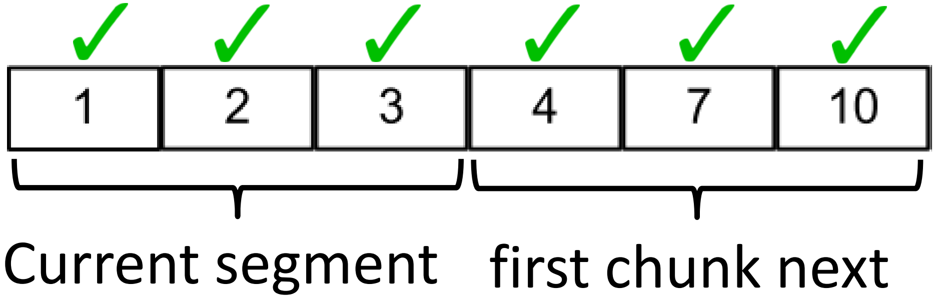
- Once branch point has been traversed, move on to next segment ...



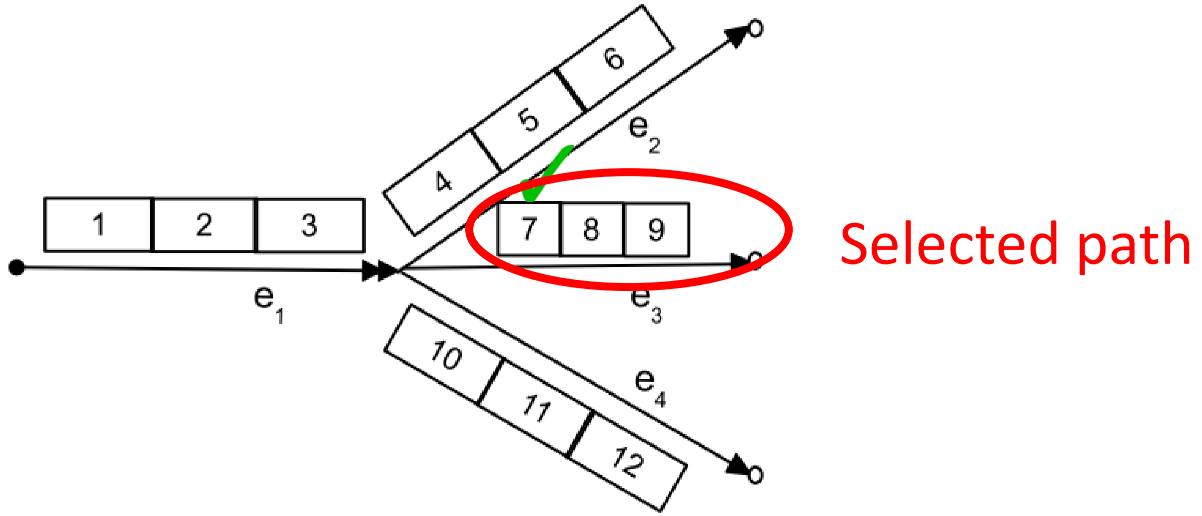
Problem Description and Constraints



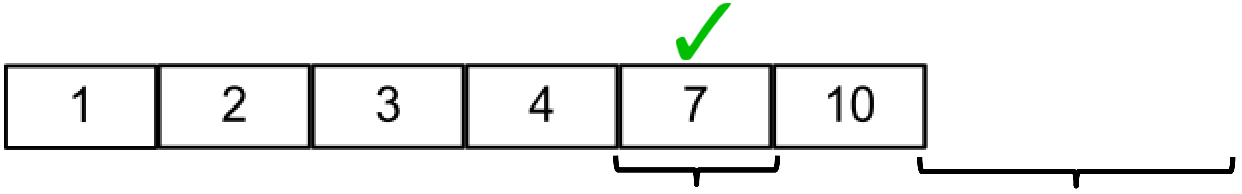
- Once branch point has been traversed, move on to next segment ...



Problem Description and Constraints

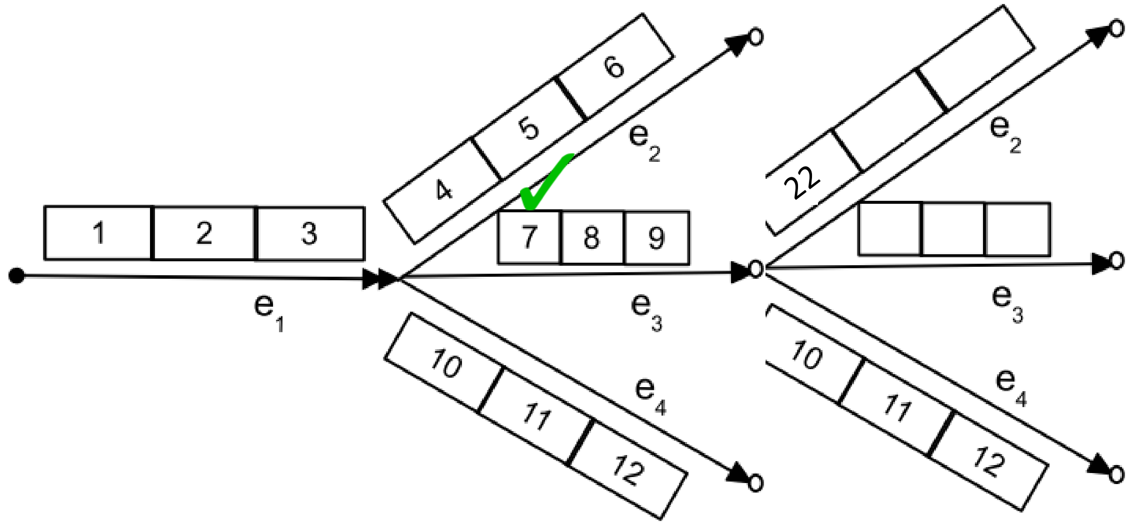


- Once branch point has been traversed, move on to next segment ...

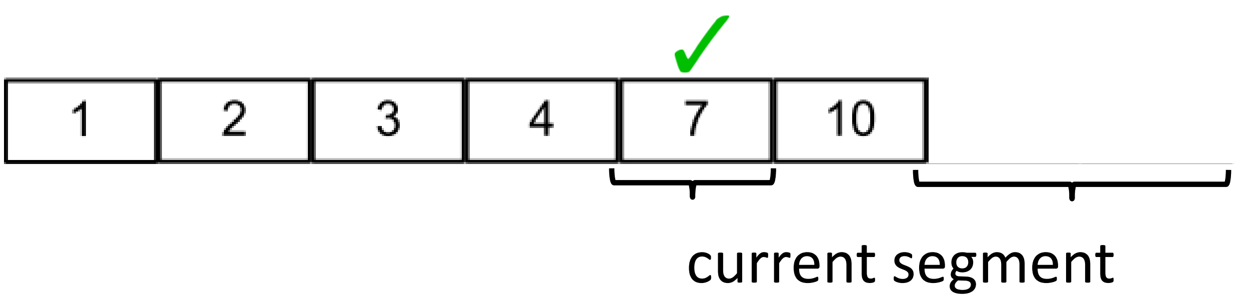


current segment

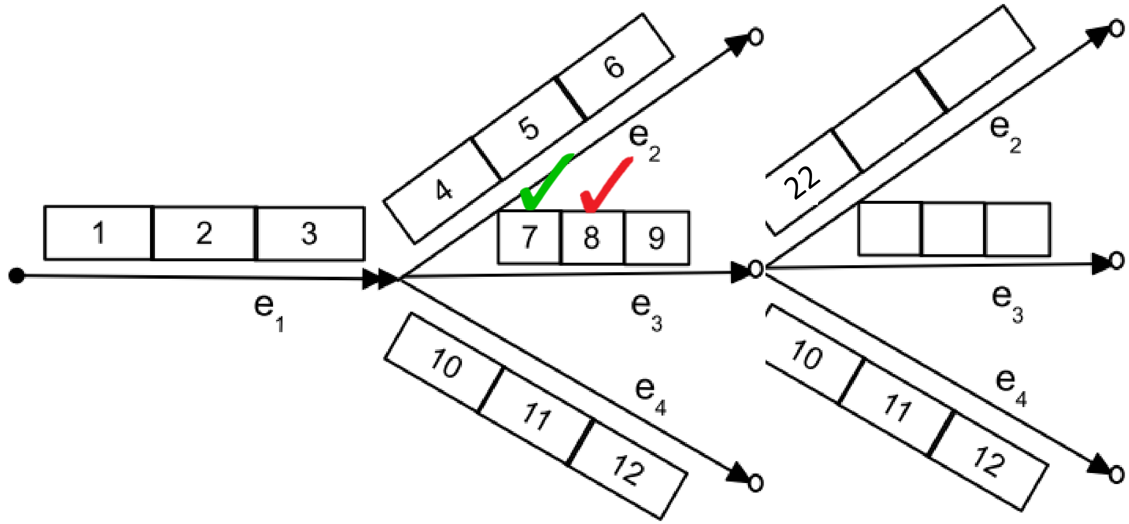
Problem Description and Constraints



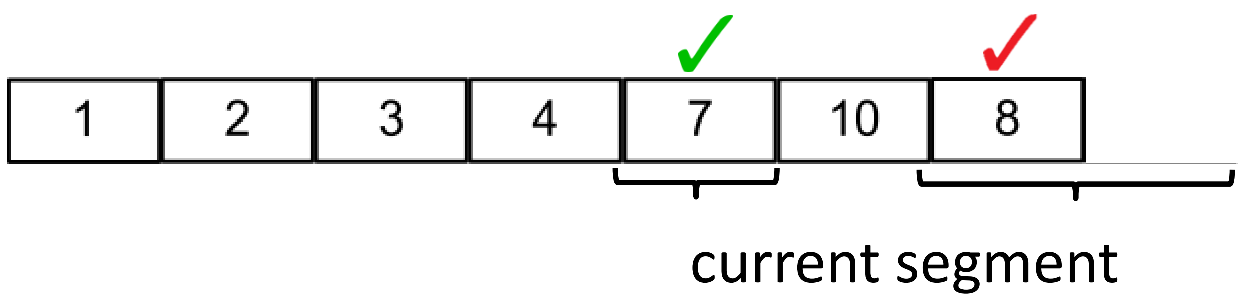
- Once branch point has been traversed, move on to next segment ...



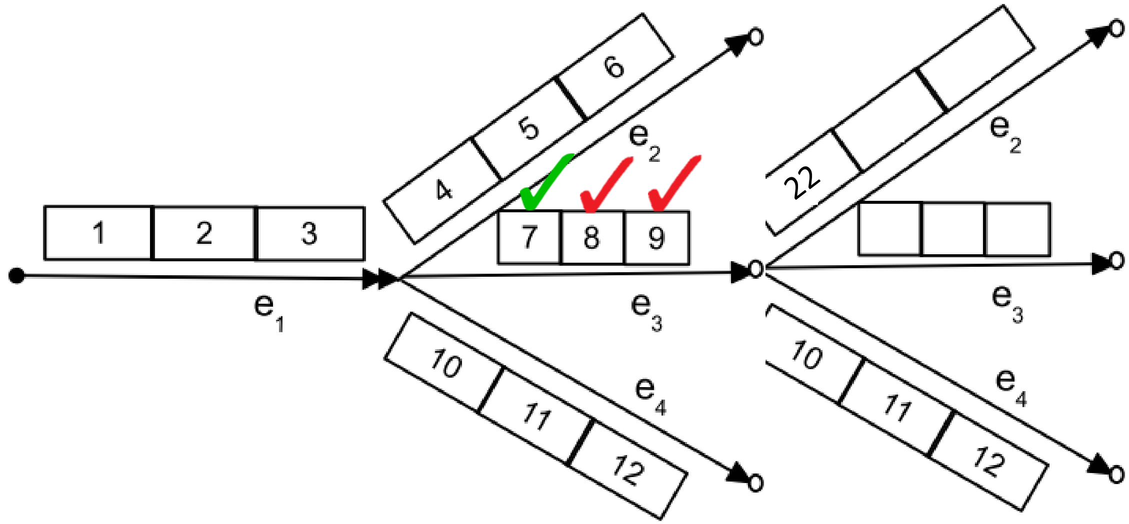
Problem Description and Constraints



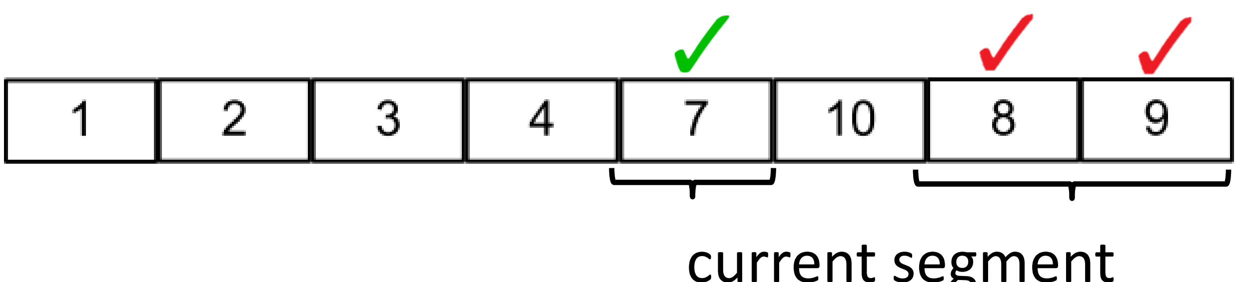
- Once branch point has been traversed, move on to next segment ...



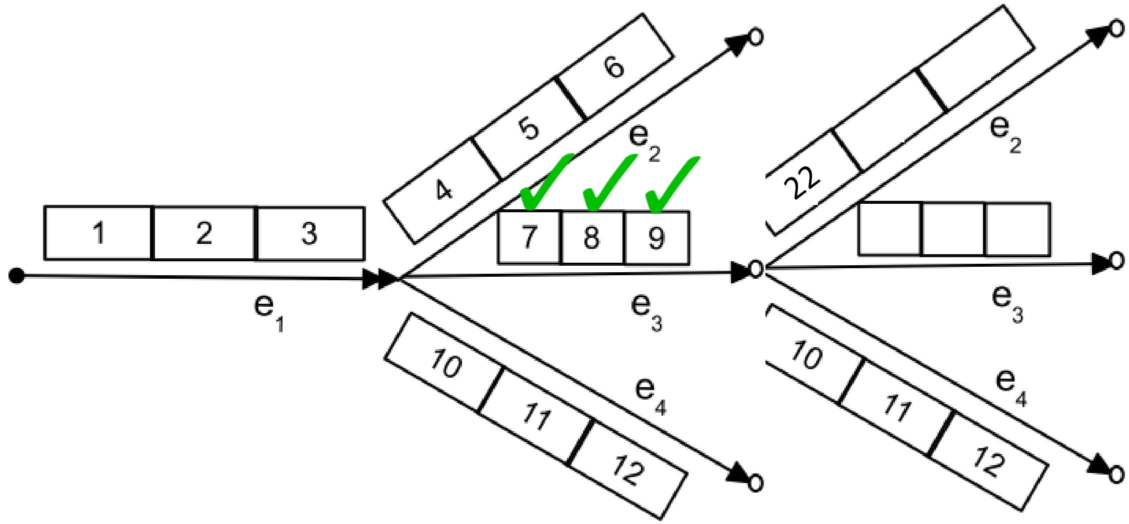
Problem Description and Constraints



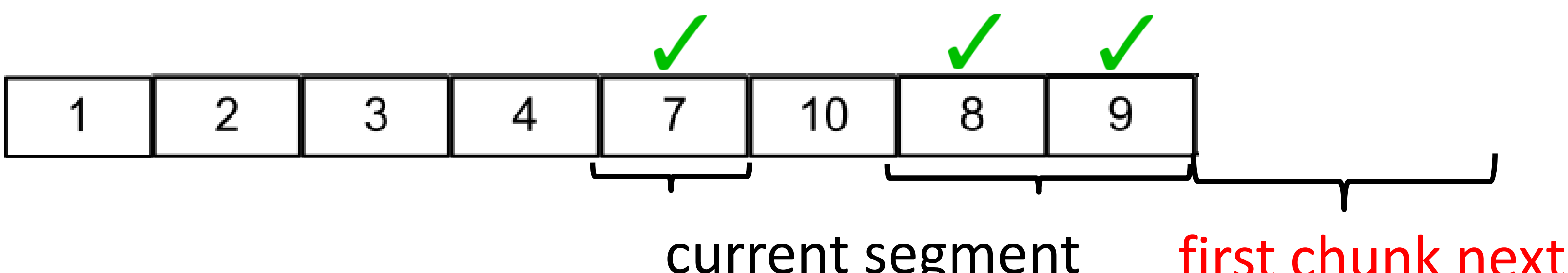
- Once branch point has been traversed, move on to next segment ...



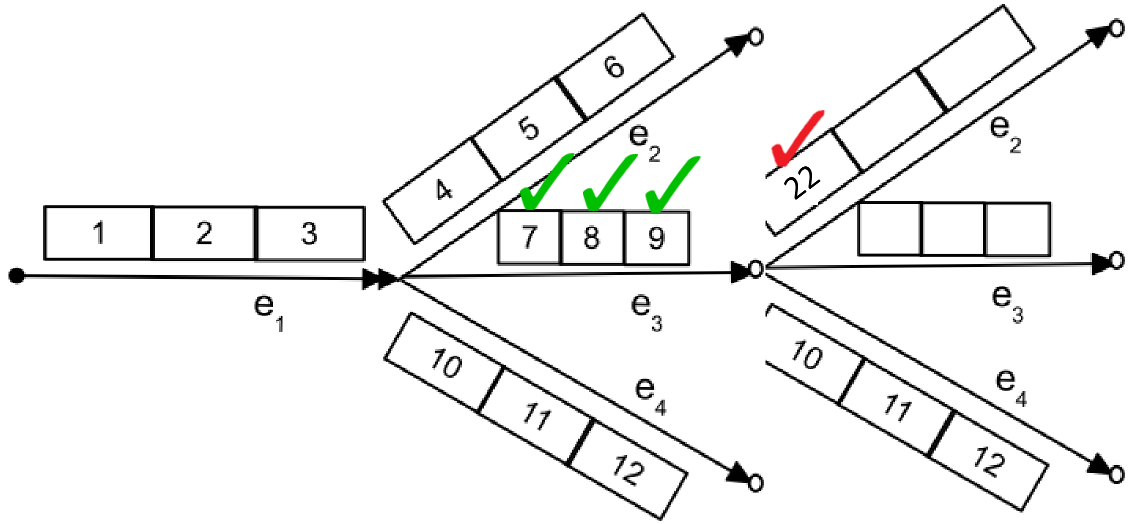
Problem Description and Constraints



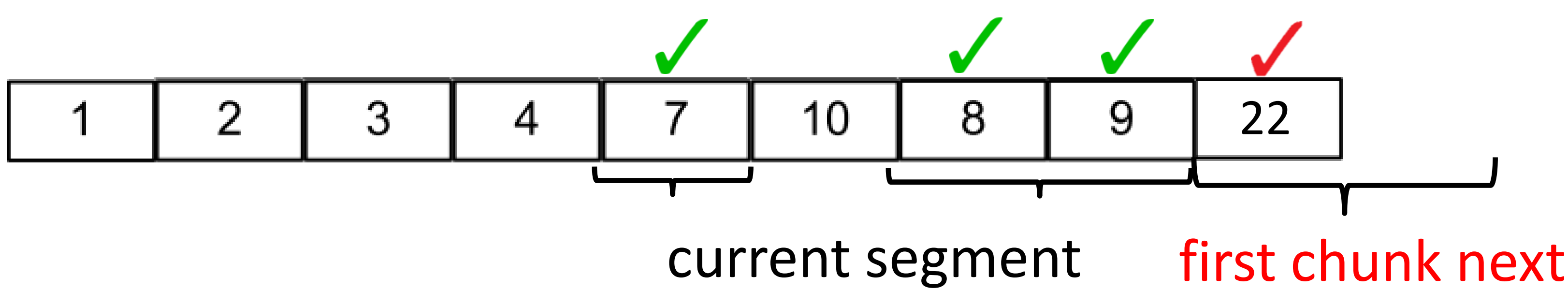
- Once branch point has been traversed, move on to next segment ...



Problem Description and Constraints

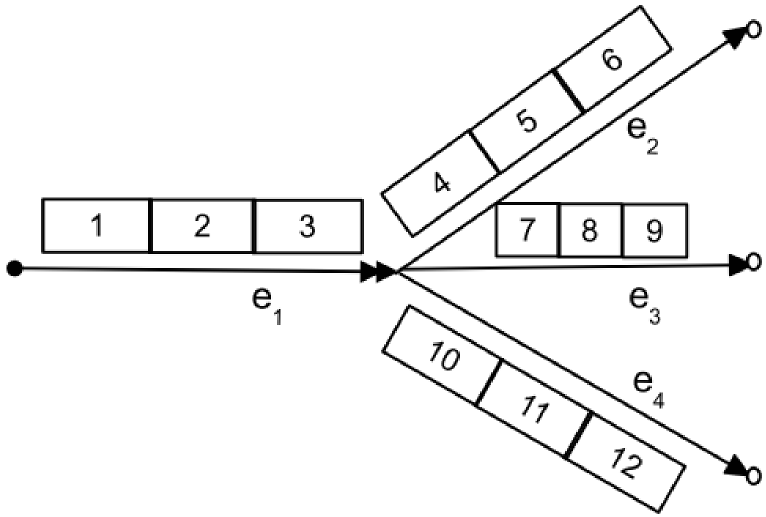


- Once branch point has been traversed, move on to next segment ...



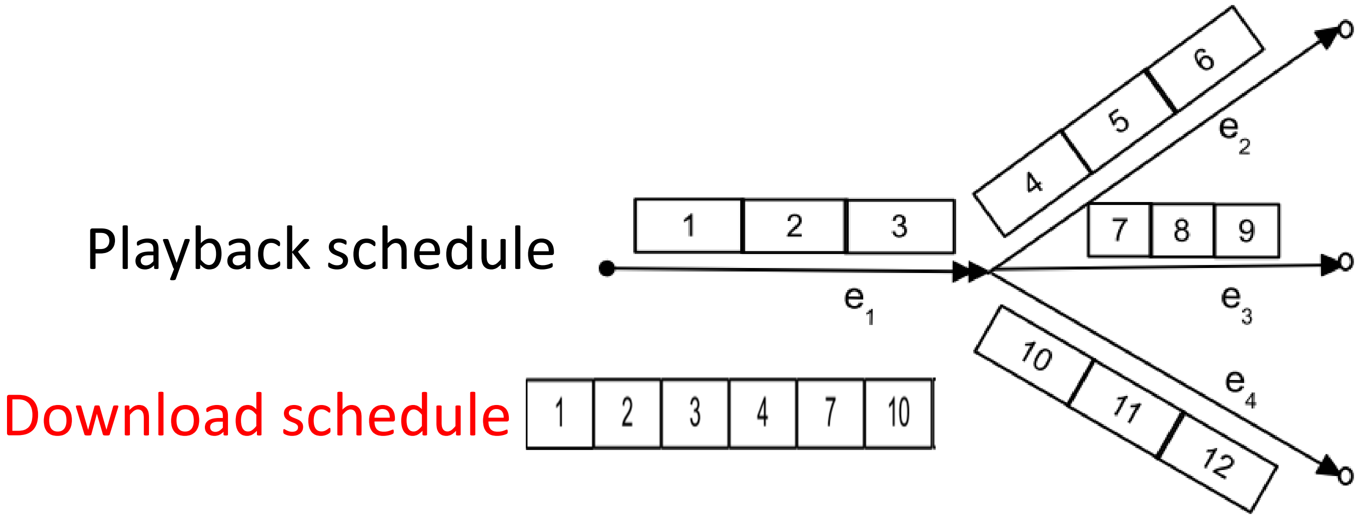
Problem Description and Constraints

Playback schedule



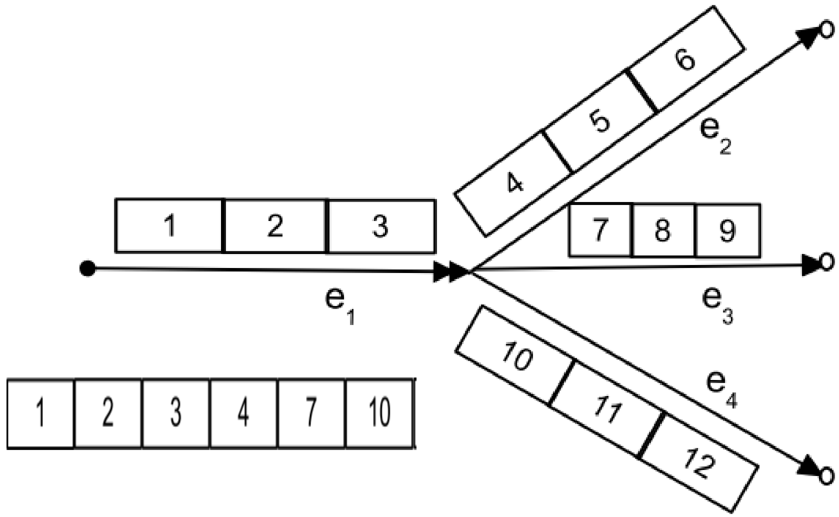
- Playback deadlines
  - for seamless playback without stalls

Problem Description and Constraints



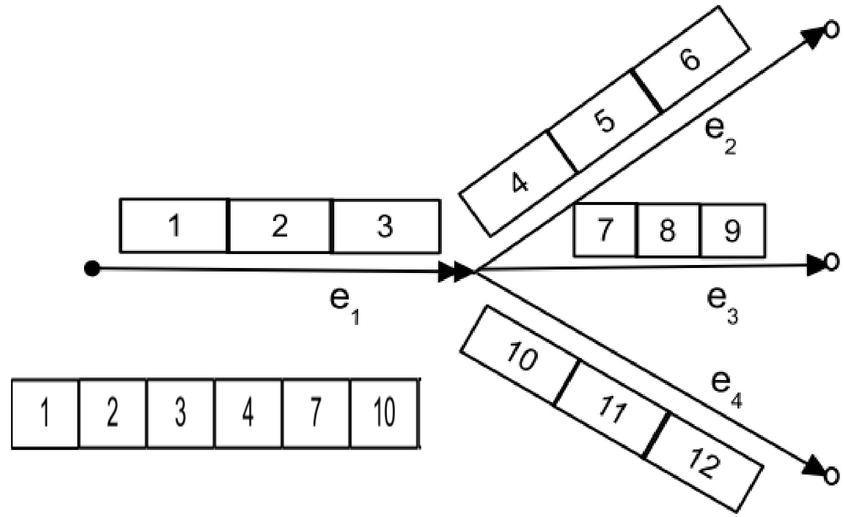
- Playback deadlines
  - for seamless playback without stalls

Problem Description and Constraints



- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

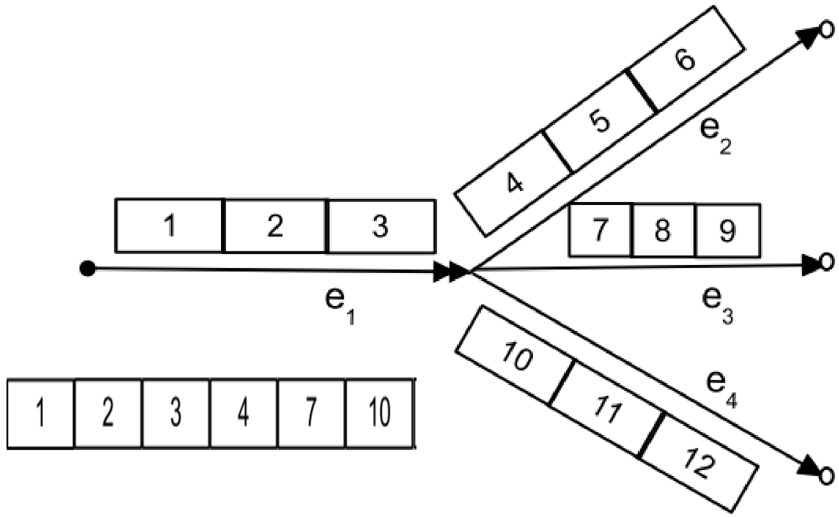
Problem Description and Constraints



- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \le i \le n_e$$

Problem Description and Constraints

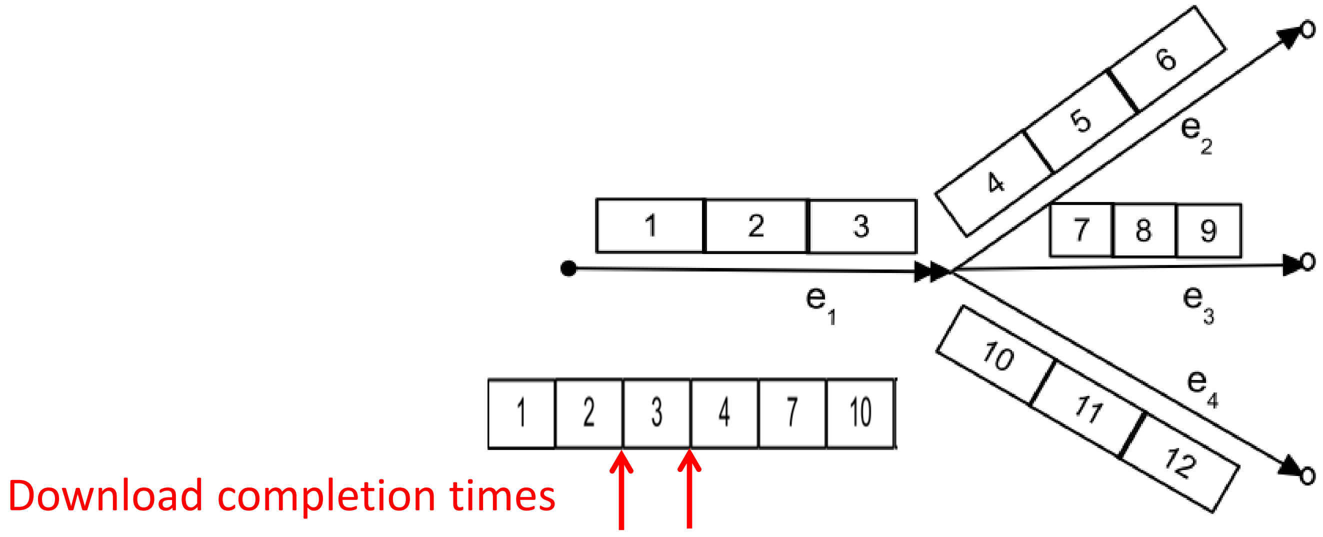


- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \le i \le n_e$$

Download completion time

Problem Description and Constraints

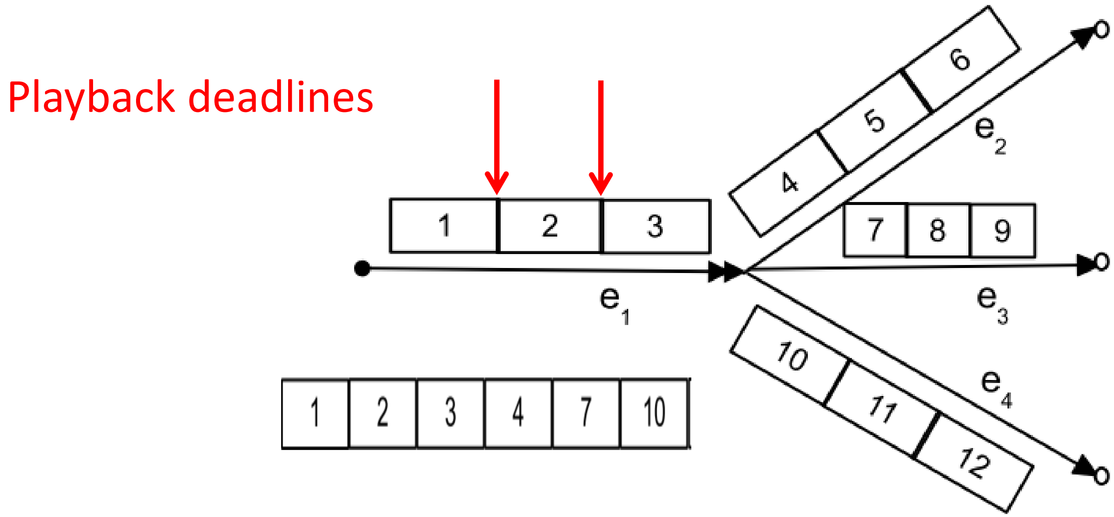


- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \leq t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \leq i \leq n_e$$

Download completion time

Problem Description and Constraints



- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

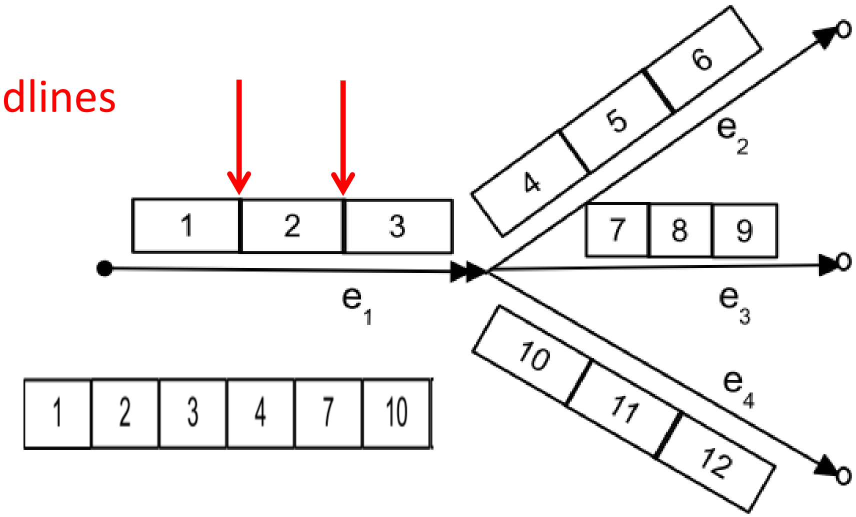
$$t_i^c \leq t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \leq i \leq n_e$$

Time of playback deadline

Download completion time

Problem Description and Constraints

Playback deadlines

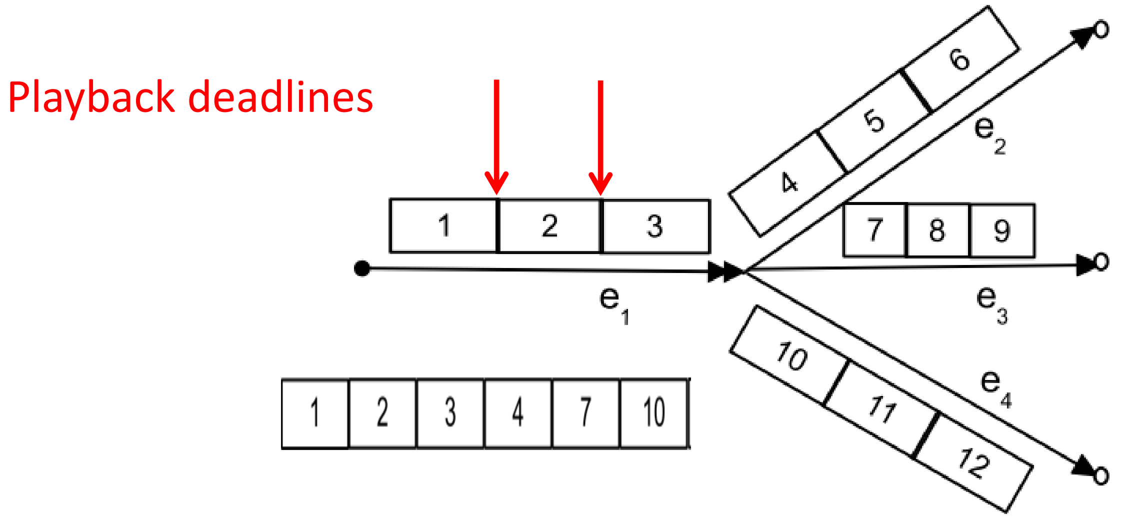


- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \le i \le n_e$$

Time of playback deadline

Problem Description and Constraints

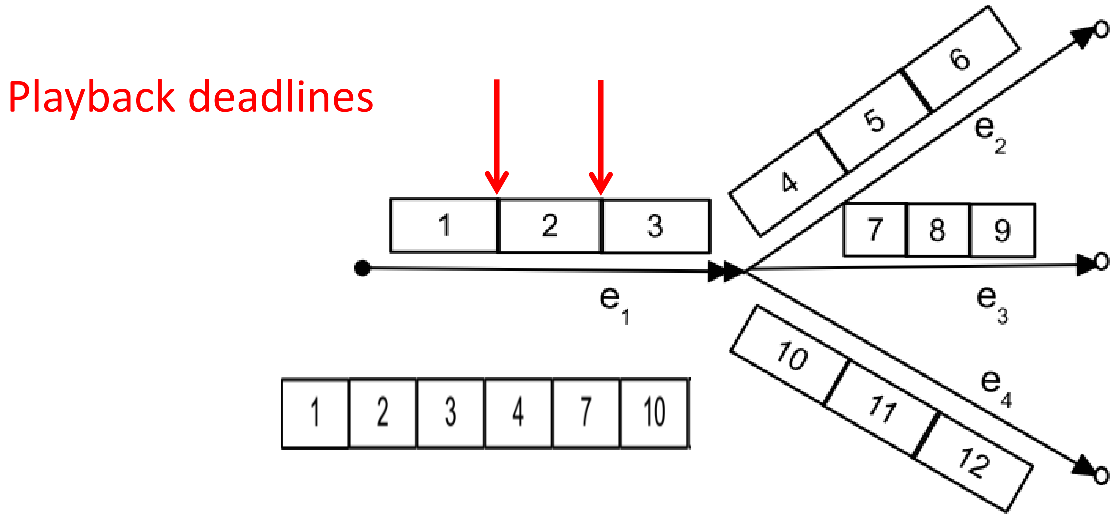


- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \le i \le n_e$$

Startup delay

Problem Description and Constraints



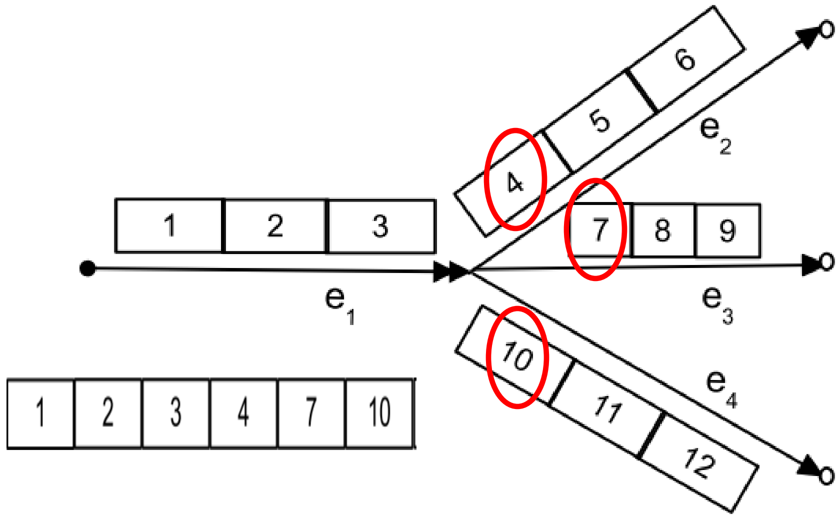
- Playback deadlines
  - for seamless playback without stalls
  - Current segment: e.g., 2 and 3

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{i-1} l_j, \quad \text{if } 1 \le i \le n_e$$

Startup delay

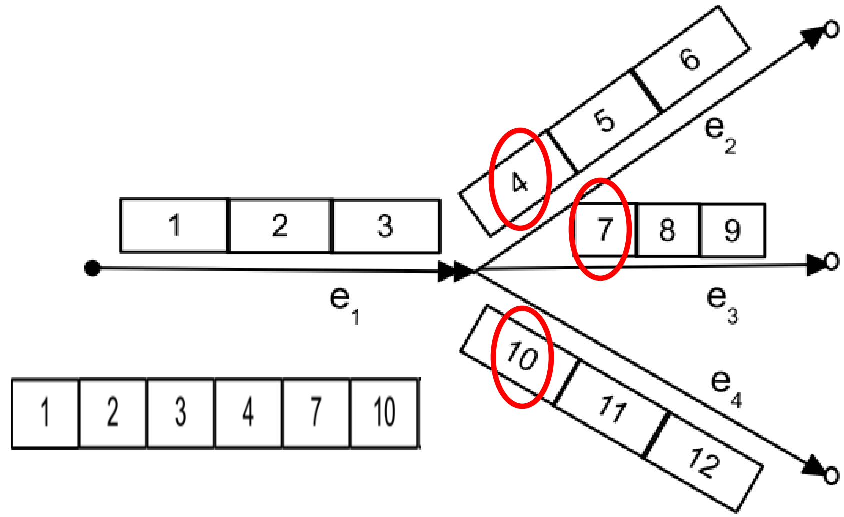
Playtime of earlier chunks

Problem Description and Constraints



- Playback deadlines
  - for seamless playback without stalls
  - First chunks next segment: e.g., 4, 7, and 10

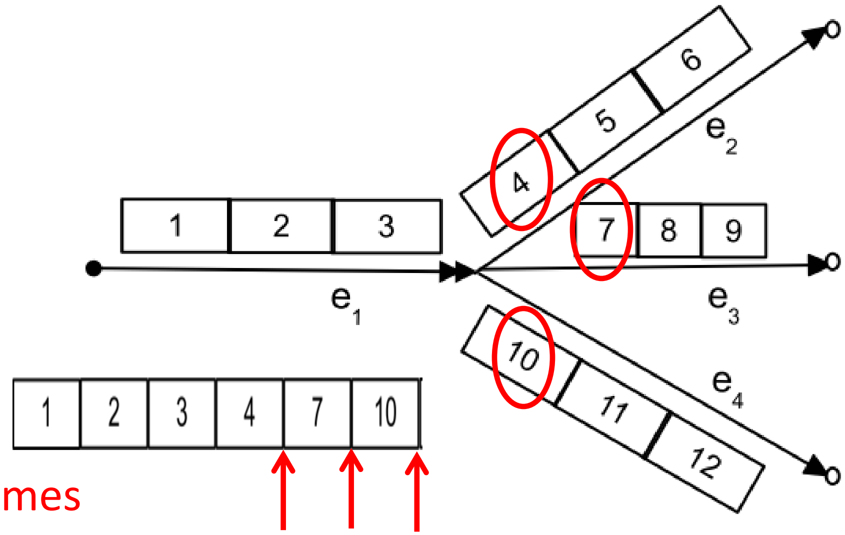
Problem Description and Constraints



- Playback deadlines
  - for seamless playback without stalls
  - First chunks next segment: e.g., 4, 7, and 10

$$t_i^c \leq t_i^d = \tau + \sum_{j=1}^{n_e} l_j, \quad \text{if } n_e < i \leq n_e + |\mathcal{E}^b|$$

Problem Description and Constraints



Download completion times

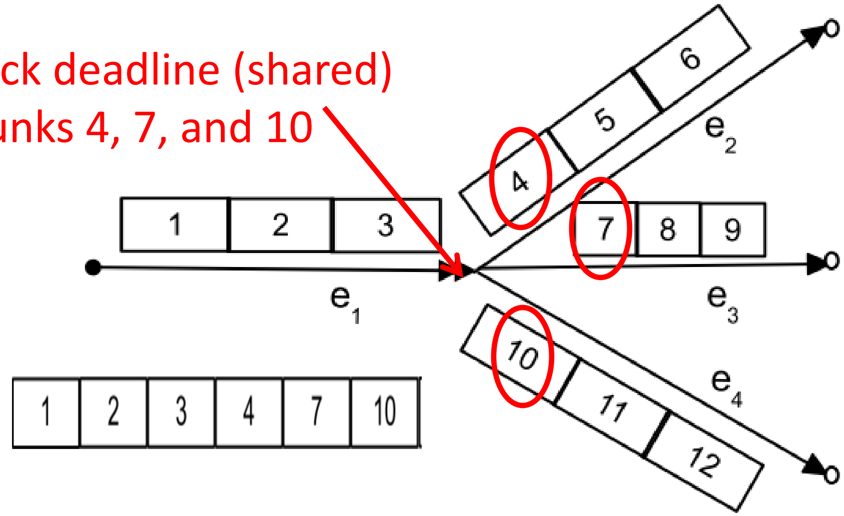
- Playback deadlines
  - for seamless playback without stalls
  - First chunks next segment: e.g., 4, 7, and 10

$$t_i^c \leq t_i^d = \tau + \sum_{j=1}^{n_e} l_j, \quad \text{if } n_e < i \leq n_e + |\mathcal{E}^b|$$

Download completion times

Problem Description and Constraints

Playback deadline (shared) for chunks 4, 7, and 10



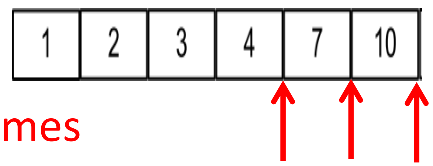
- Playback deadlines
  - for seamless playback without stalls
  - First chunks next segment: e.g., 4, 7, and 10

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{n_e} l_j, \quad \text{if } n_e < i \le n_e + |\mathcal{E}^b|$$

Time at which branch point is reached

Download completion times

Problem Description and Constraints



Download completion times

$$t_i^c \le t_i^d = \tau + \sum_{j=1}^{n_e} l_j, \quad \text{if } n_e < i \le n_e + |\mathcal{E}^b|$$

Download completion times

Problem Description and Constraints

- Download times t_i^c , rate estimations, and parallel connections

Problem Description and Constraints

- Download times t_i^c, rate estimations, and parallel connections
  - At the end of a chunk download, schedule new downloads and new TCP connections
  - Assume that an additional TCP connection will not increase the total download rate
  - New connections are initiated only if it is not expected to lead to playback deadline violations

Problem Description and Constraints

- Download times t_i^c , rate estimations, and parallel connections
  - At the end of a chunk download, schedule new downloads and new TCP connections
  - Assume that an additional TCP connection will not increase the total download rate
  - New connections are initiated only if it is not expected to lead to playback deadline violations

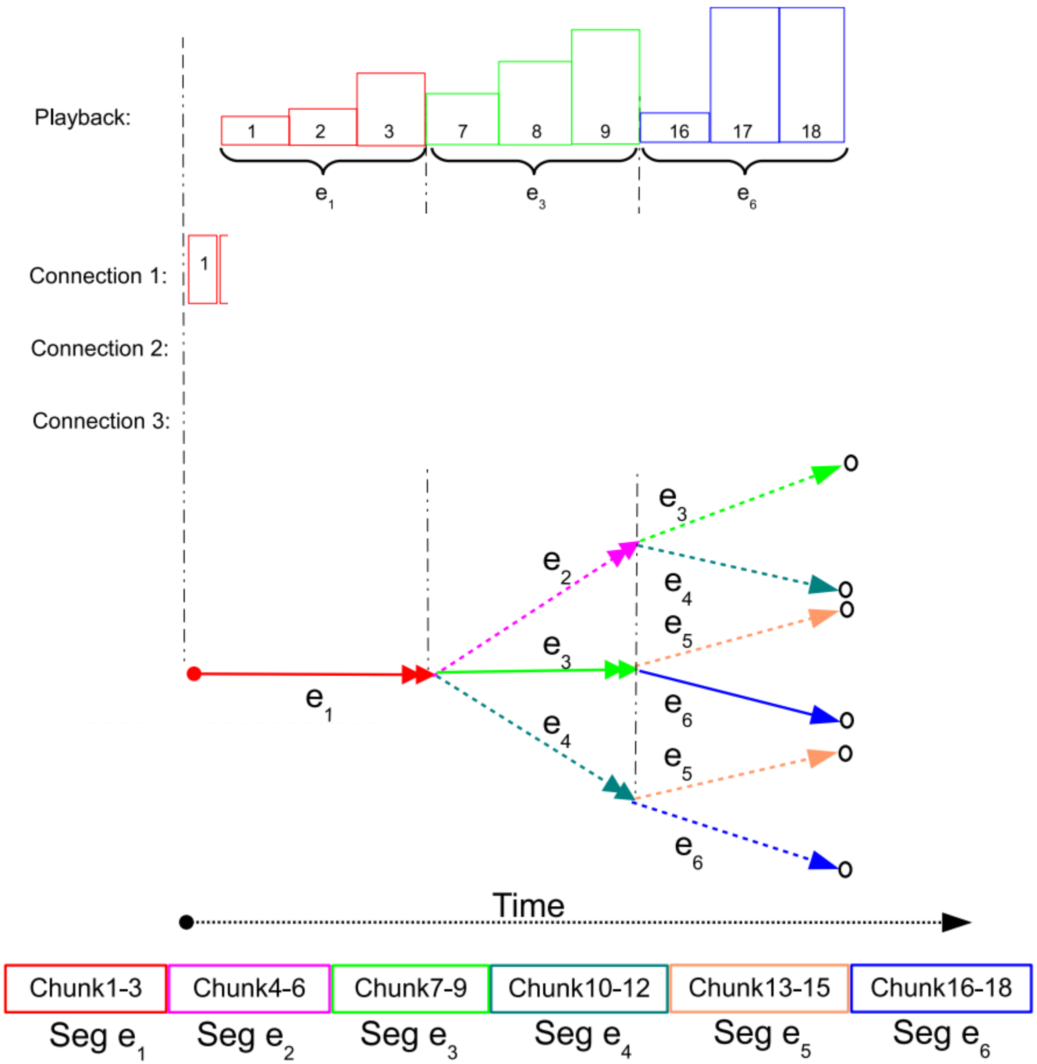
Problem Description and Constraints

- Download times t_i^c , rate estimations, and parallel connections
  - At the end of a chunk download, schedule new downloads and new TCP connections
  - Assume that an additional TCP connection will not increase the total download rate
  - New connections are initiated only if it is not expected to lead to playback deadline violations

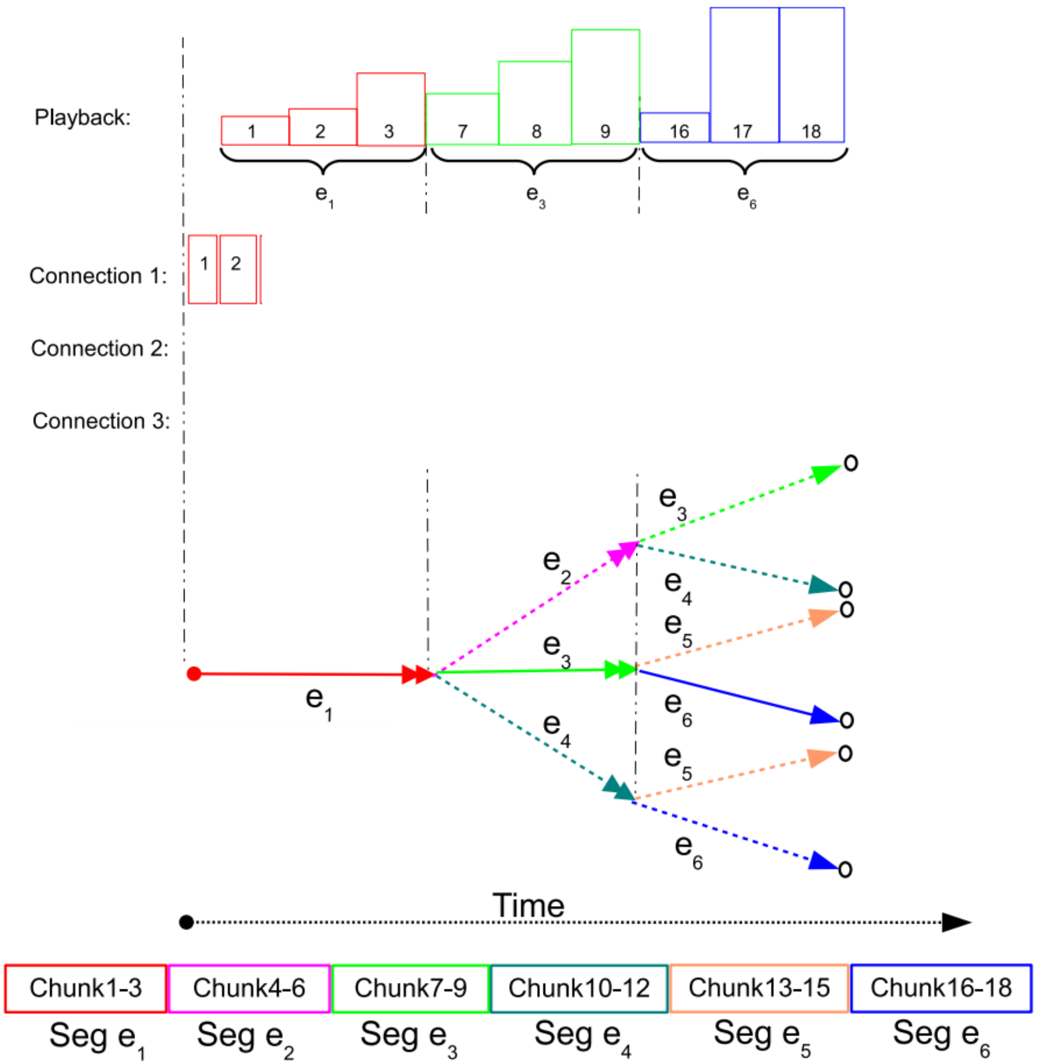
Problem Description and Constraints

- Download times t_i^c , rate estimations, and parallel connections
  - At the end of a chunk download, schedule new downloads and new TCP connections
  - Assume that an additional TCP connection will not increase the total download rate
  - New connections are initiated only if it is not expected to lead to playback deadline violations

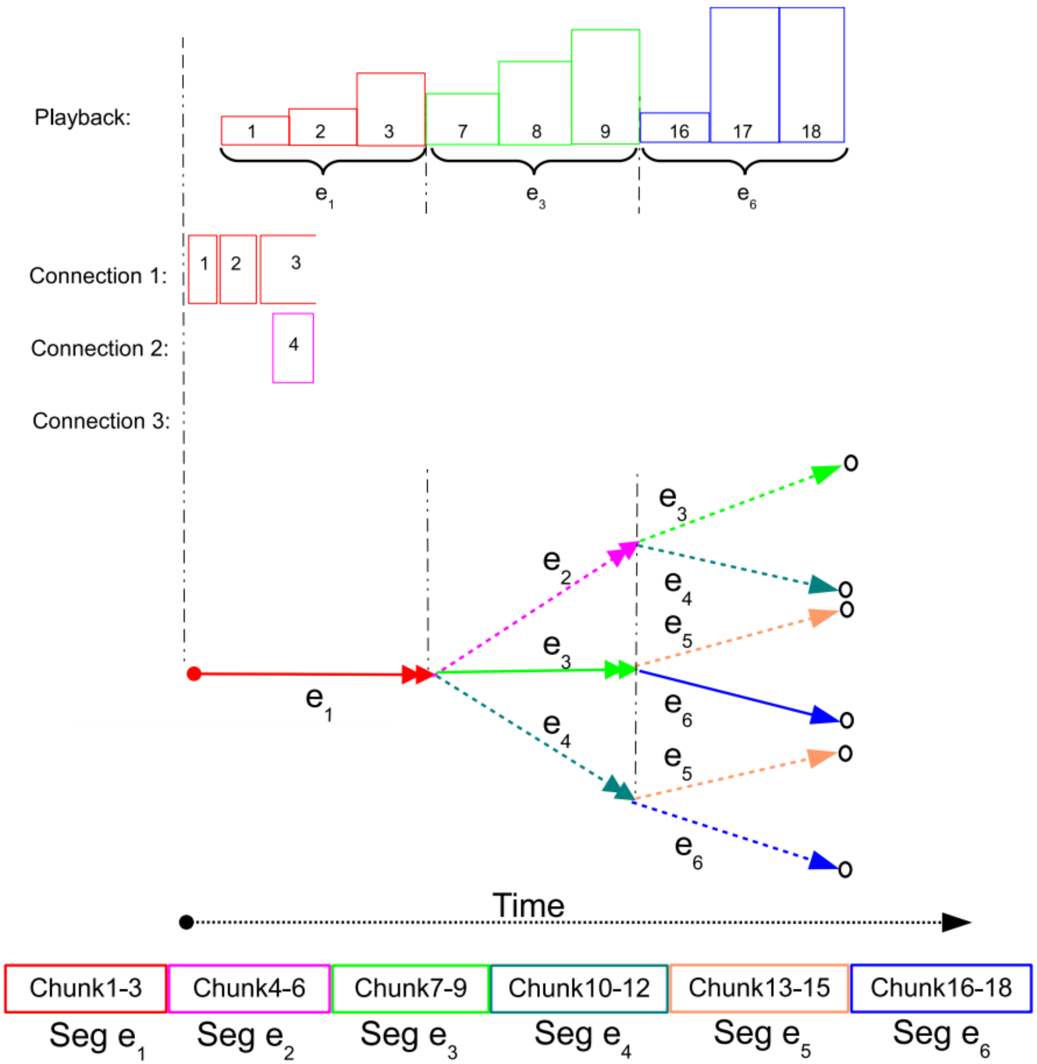
Concurrent Download Example



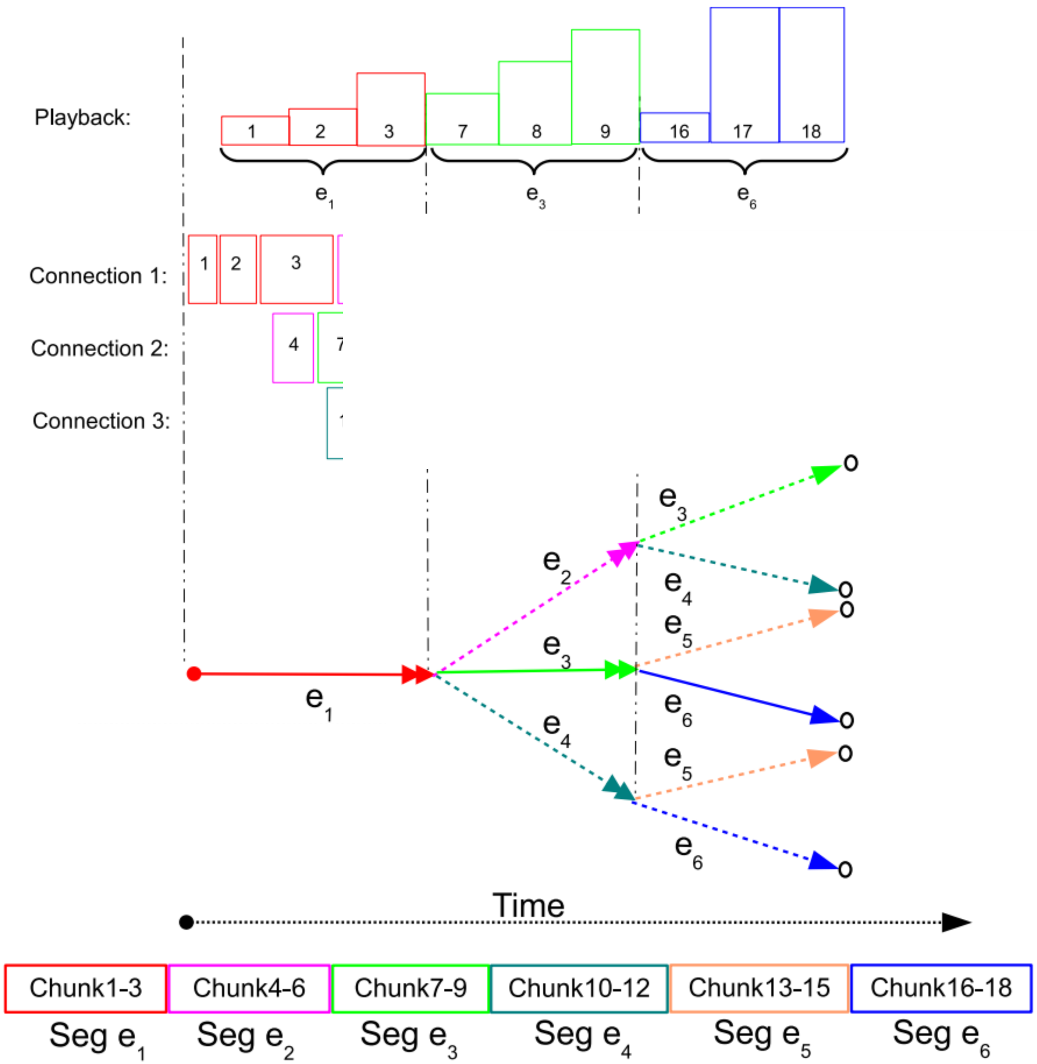
Concurrent Download Example



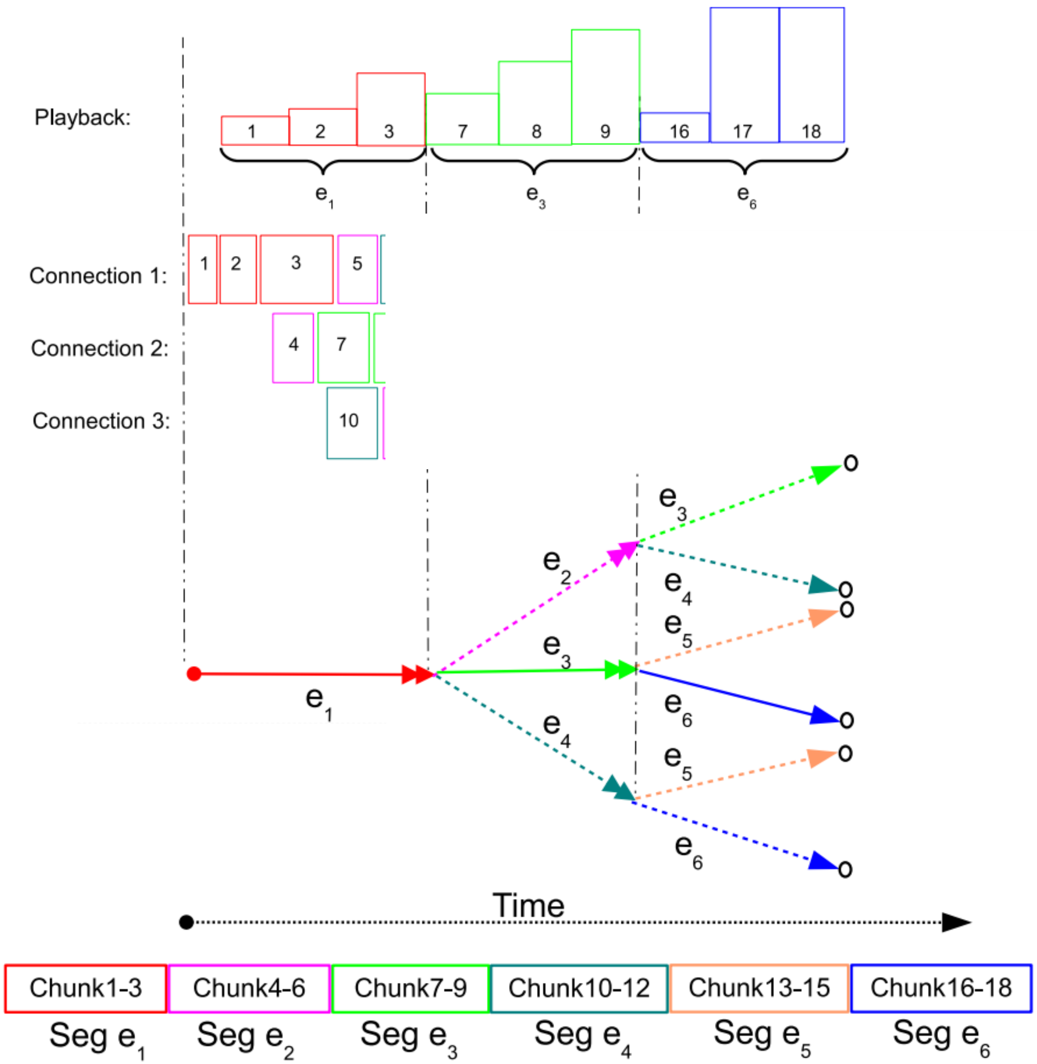
Concurrent Download Example



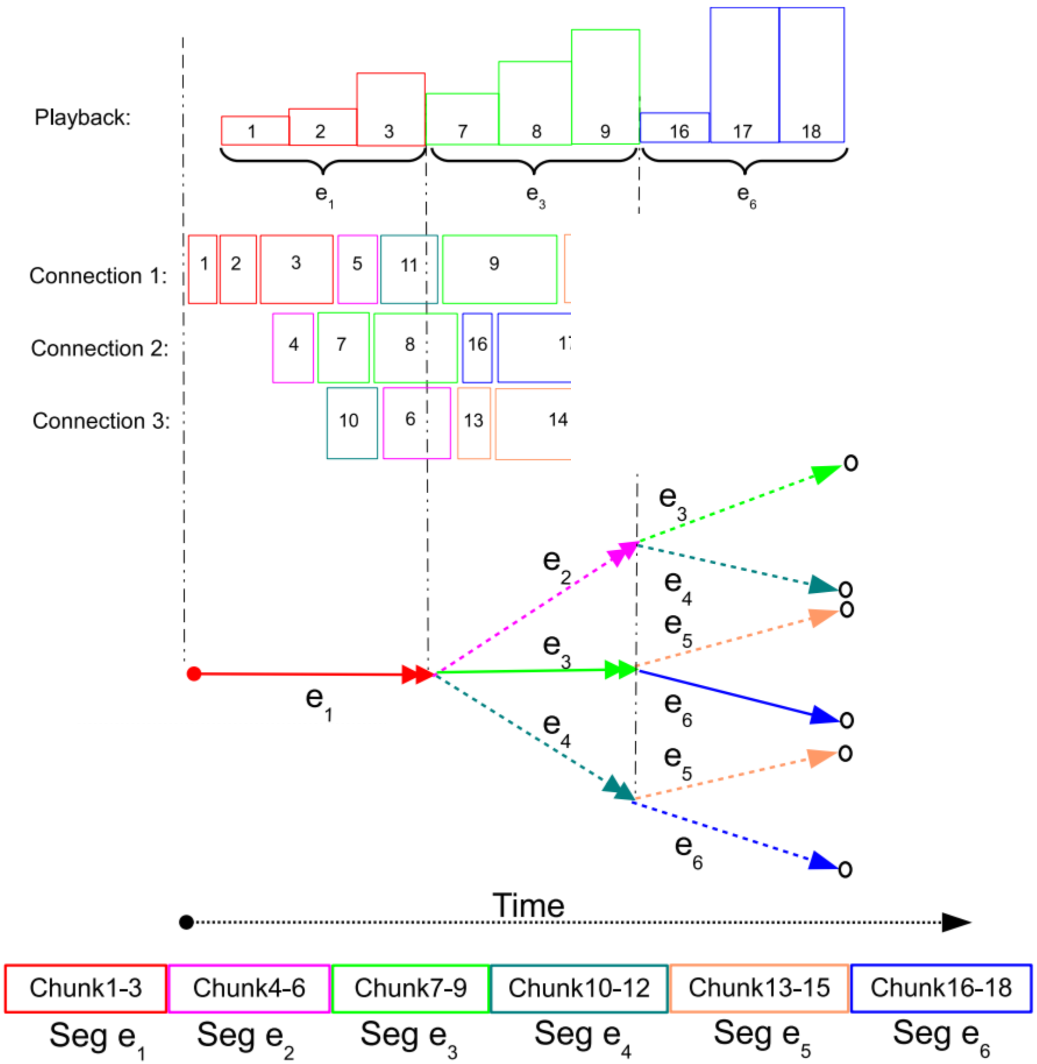
Concurrent Download Example



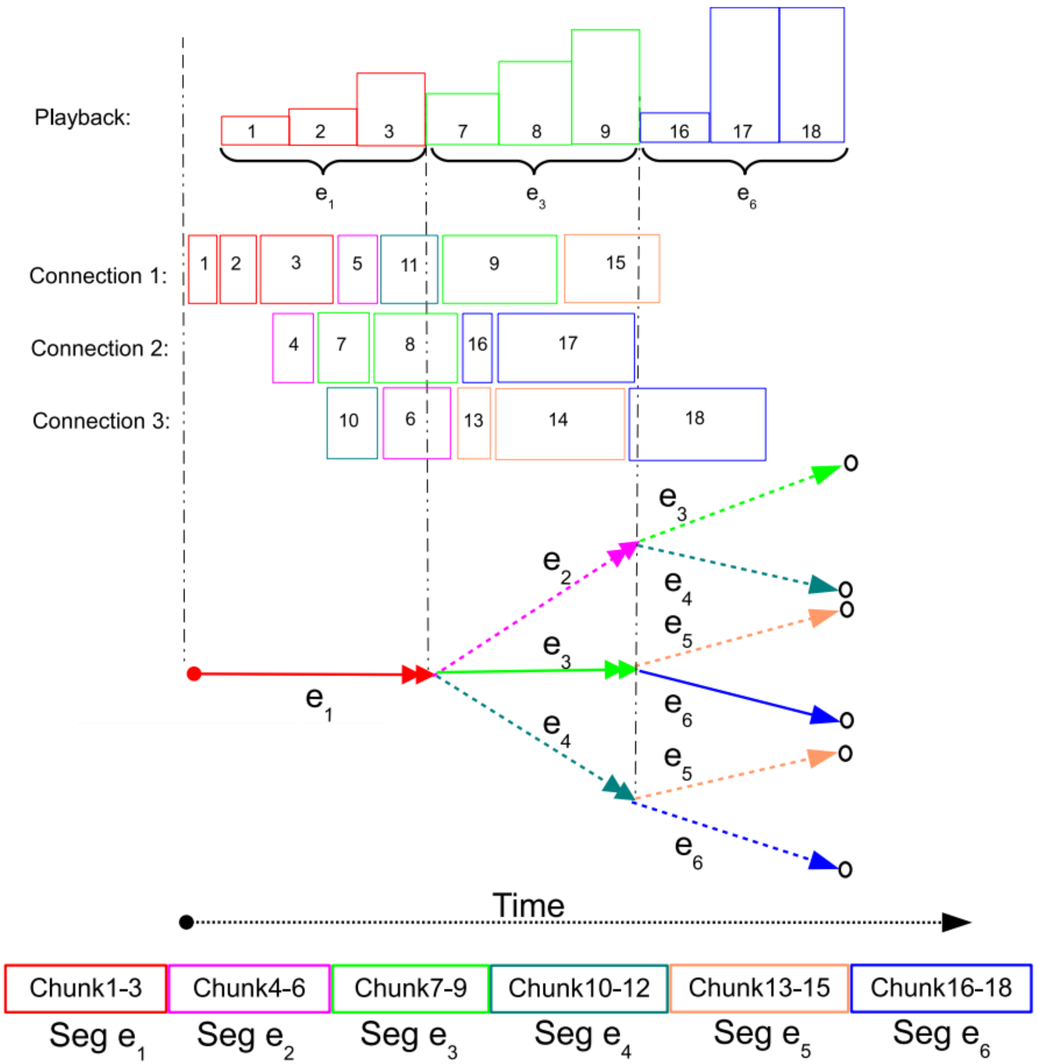
Concurrent Download Example



Concurrent Download Example



Concurrent Download Example



Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback

Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback

Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback

Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback

Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback
- Exponential number of candidate schedules

Prefetching Policies

- At download completion
  - Decide number of chunks to download next
  - Decide quality level of chunks
  - Maximize expected weighted playback
- Exponential number of candidate schedules
- Our optimized policies restrict the number of candidate schedules to consider
  - Policies differ in number of candidate schedules and how aggressive they are (in choosing qualities)

Comparison Between Policies

Policy	Connections	Schedules considered	Objective
All schedules	$1 \le c_i \le C^{max}$	Q^M, where $M=n_e+\|\xi_b\|-m$	-
Optimized non-increasing quality	$1 \le c_i \le C^{max}$	$\binom{M+Q-1}{Q-1}$	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\|\xi_b\|} q_i l_i$
Optimized maintainable quality	$1 \le c_i \le C^{max}$	Q	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\|\xi_b\|} q_i l_i$

- Total number of schedules: Q^M
- Optimized non-increasing quality:
  - Constraint: Qualities of consecutive chunks are non-increasing
- Optimized maintainable quality:
  - Constraint: Chosen quality must be sustainable for the remaining chunks

Comparison Between Policies

Policy	Connections	Schedules considered	Objective
All schedules	$1 \leq c_i \leq C^{max}$	Q^M, where $M = n_e + \lvert \xi_b \rvert - m$	-
Optimized non-increasing quality	$1 \leq c_i \leq C^{max}$	$\binom{M+Q-1}{Q-1}$	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\lvert \xi_b \rvert} q_i l_i$
Optimized maintainable quality	$1 \leq c_i \leq C^{max}$	Q	

- Total number of schedules: Q^M
- Optimized non-increasing quality:
  - Constraint: Qualities of consecutive chunks are non-increasing
- Optimized maintainable quality:
  - Constraint: Chosen quality must be sustainable for the remaining chunks

Comparison Between Policies

Policy	Connections	Schedules considered	Objective
All schedules	$1 \le c_i \le C^{max}$	Q^M, where $M = n_e + \lvert \xi_b \rvert - m$	-
Optimized non-increasing quality	$1 \le c_i \le C^{max}$	$\binom{M+Q-1}{Q-1}$	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\lvert \xi_b \rvert} q_i l_i$
Optimized maintainable quality	$1 \le c_i \le C^{max}$	Q	

- Total number of schedules: Q^M
- Optimized non-increasing quality:
  - Constraint: Qualities of consecutive chunks are non-increasing
- Optimized maintainable quality:
  - Constraint: Chosen quality must be sustainable for the remaining chunks

Comparison Between Policies

Policy	Connections	Schedules considered	Objective
Single connection	1	Q	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\lvert\xi_b\rvert} q_i l_i$
Greedy bandwidth	$1 \le c_i \le C^{max}$		$\sum_{i=j}^{j+m} q_i l_i$

- Single connection: baseline comparing to policies which do not use multiple connections
- Greedy bandwidth: bandwidth aggressive as opposed to aggressive quality choices
- Naïve: benchmark to regular branched video players

Comparison Between Policies

Policy	Connections	Schedules considered	Objective
Single connection	1	Q	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\lvert\xi_b\rvert} q_i l_i$
Greedy bandwidth	$1 \le c_i \le C^{max}$		$\sum_{i=j}^{j+m} q_i l_i$

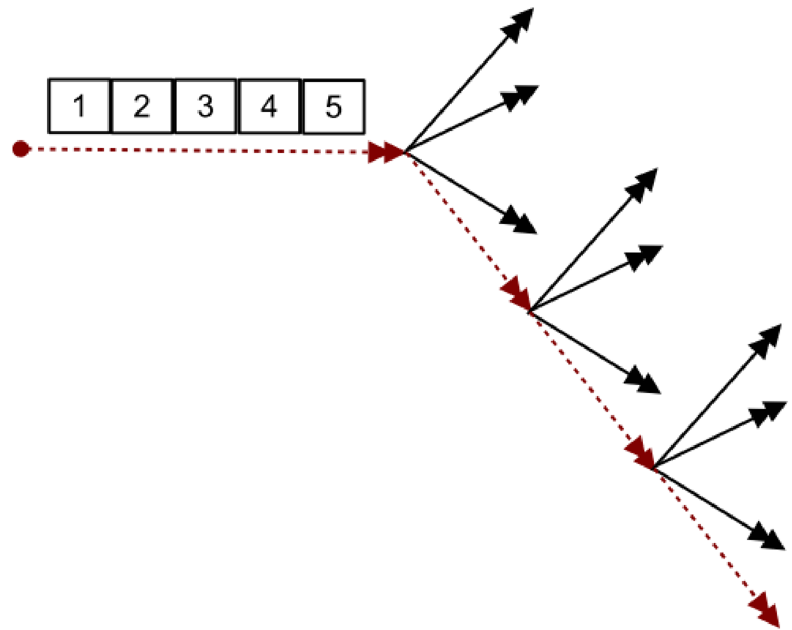
- Single connection: baseline comparing to policies which do not use multiple connections
- Greedy bandwidth: bandwidth aggressive as opposed to aggressive quality choices
- Naïve: benchmark to regular branched video players

Comparison Between Policies

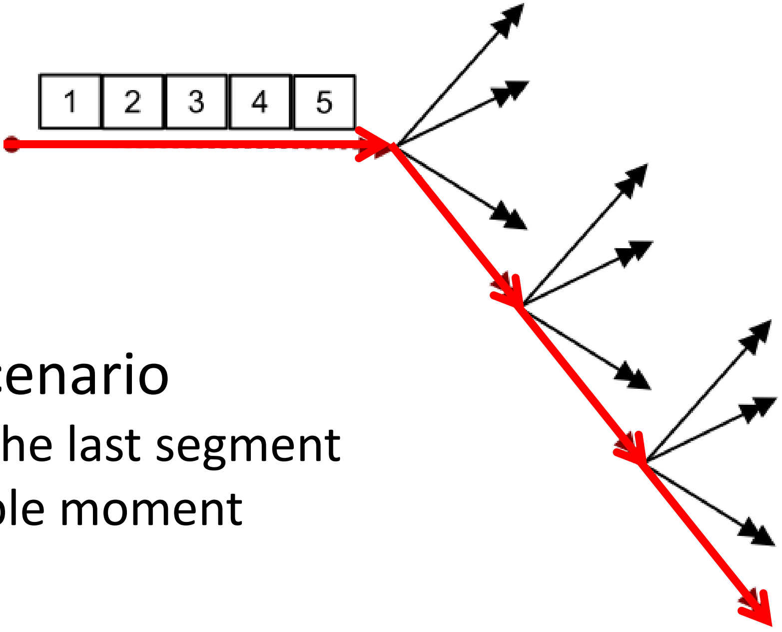
Policy	Connections	Schedules considered	Objective
Single connection	1	Q	$\sum_{i=1}^{n_e} q_i l_i + \sum_{i=n_e+1}^{n_e+\lvert\xi_b\rvert} q_i l_i$
Greedy bandwidth	$1 \le c_i \le C^{max}$		$\sum_{i=j}^{j+m} q_i l_i$

- Single connection: baseline comparing to policies which do not use multiple connections
- Greedy bandwidth: bandwidth aggressive as opposed to aggressive quality choices
- Naïve: benchmark to regular branched video players

Test Scenario



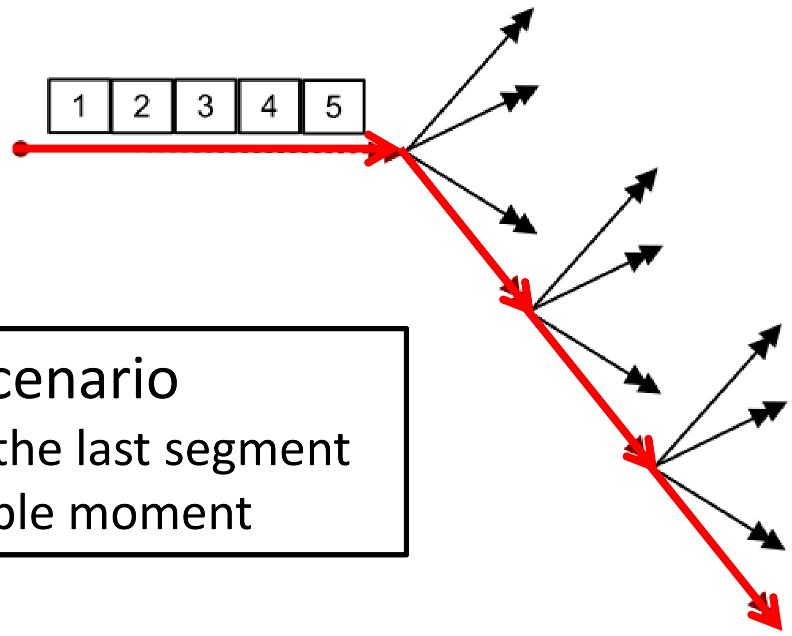
Test Scenario



Worst case scenario

- always pick the last segment
- at last possible moment

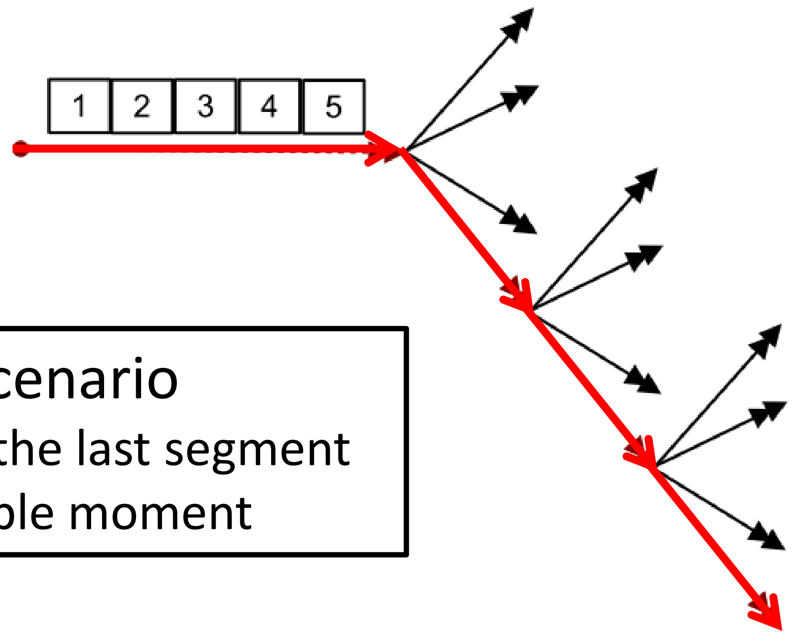
Test Scenario



Worst case scenario

- always pick the last segment
- at last possible moment

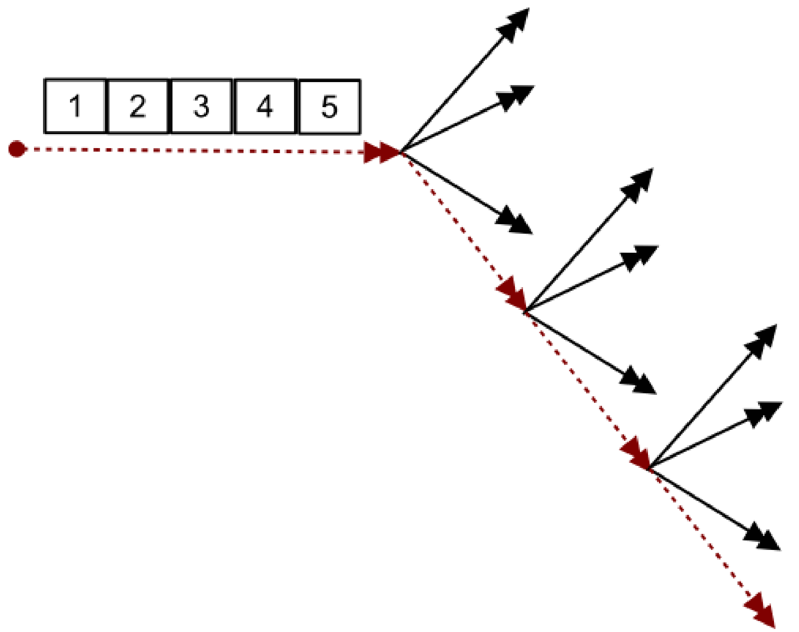
Test Scenario



Worst case scenario
- always pick the last segment
- at last possible moment

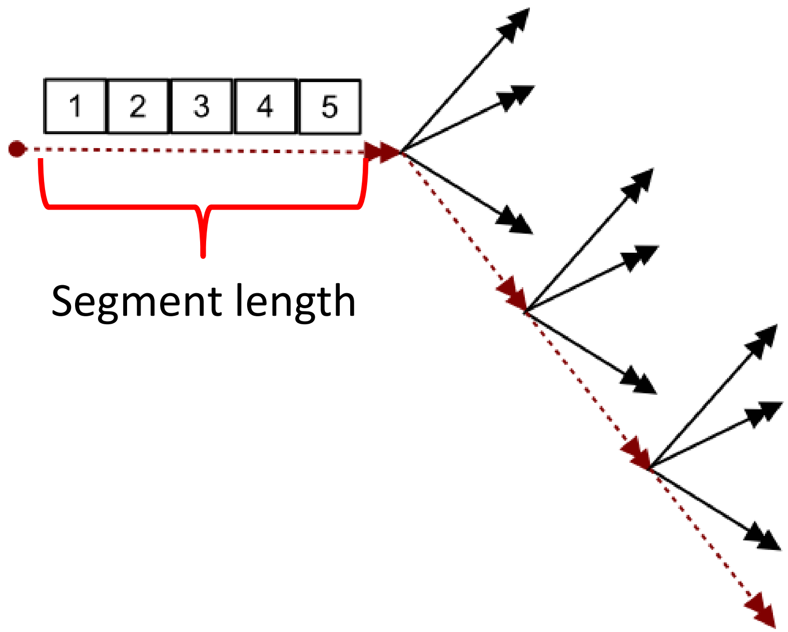
- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3

Test Scenario



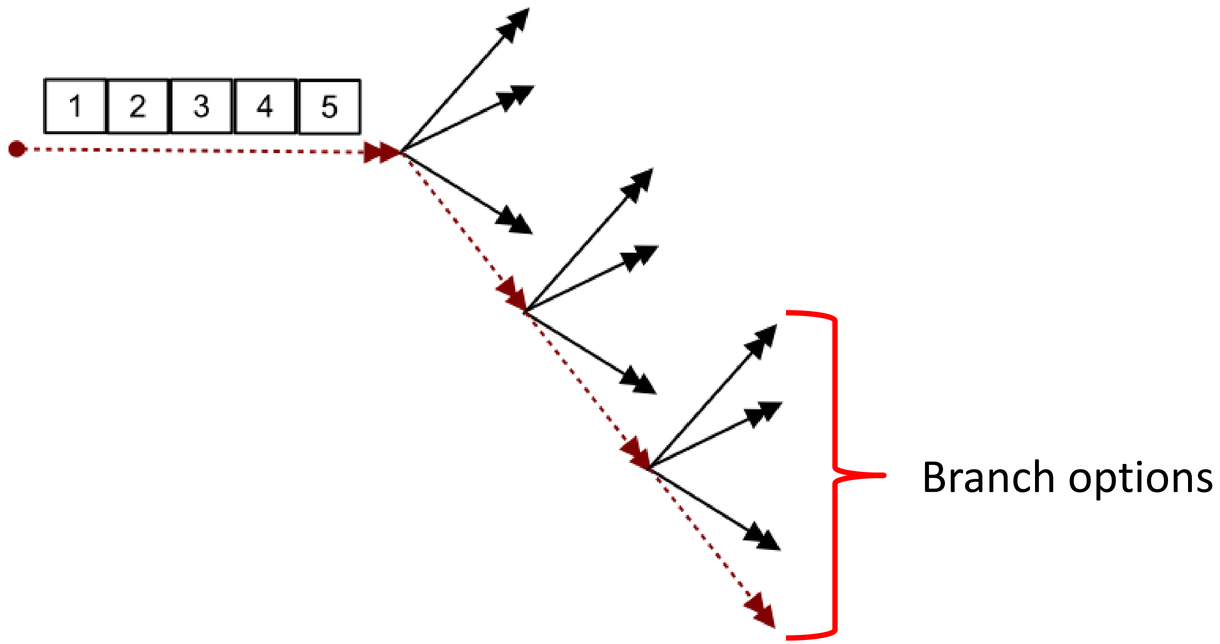
- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3

Test Scenario



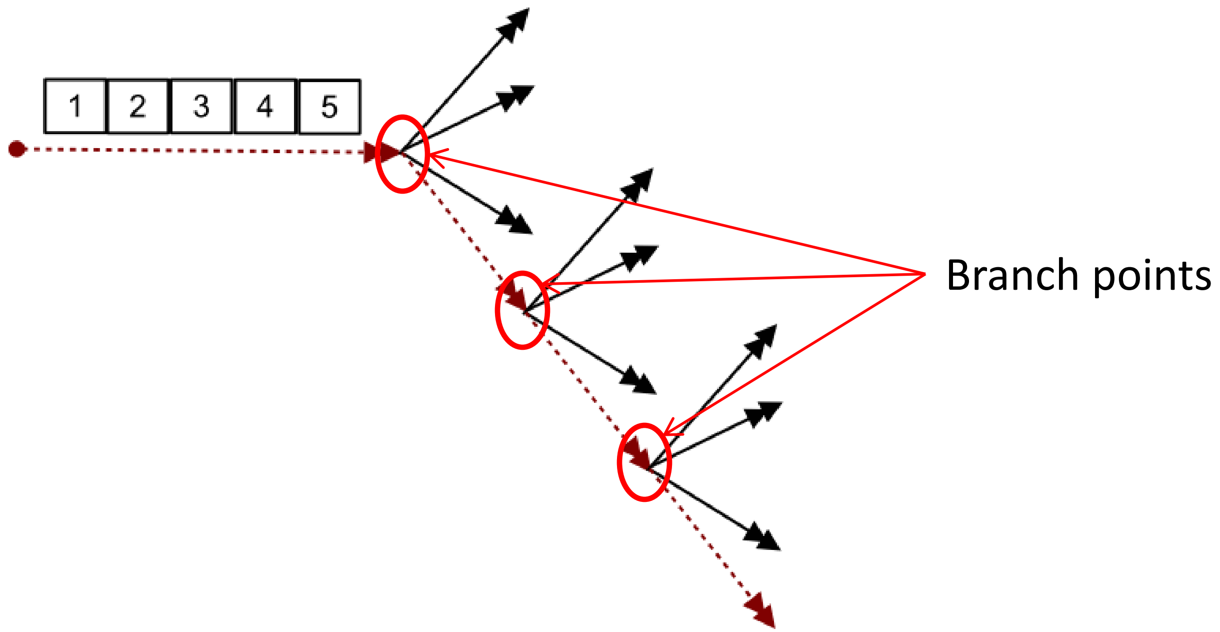
- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3

Test Scenario



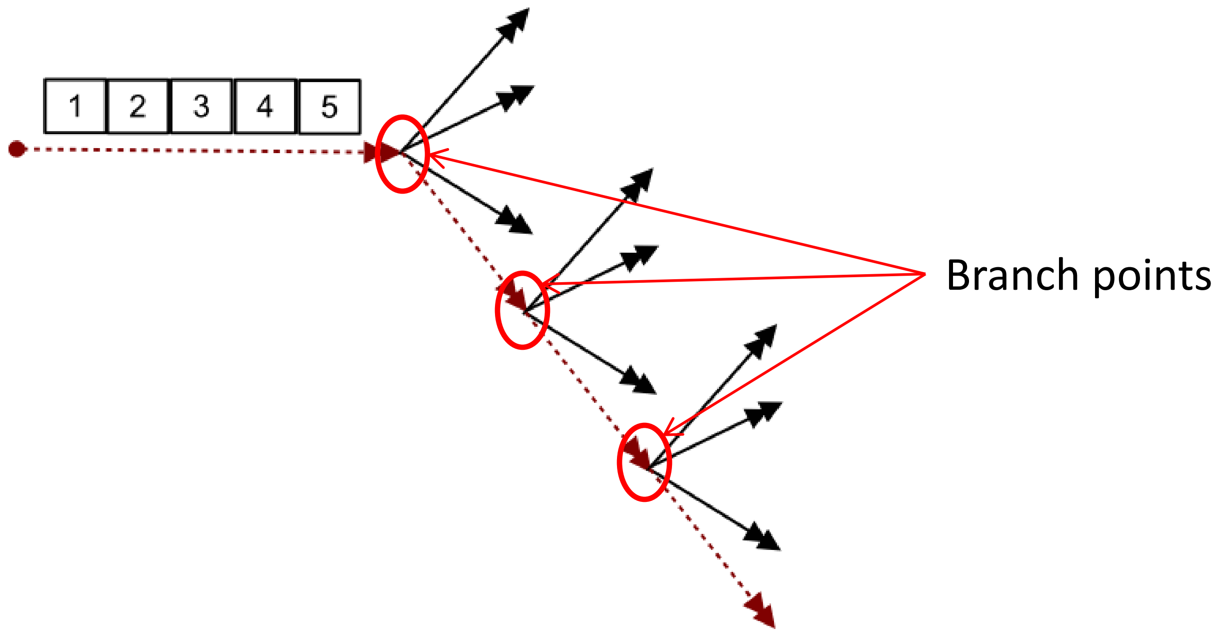
- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3

Test Scenario



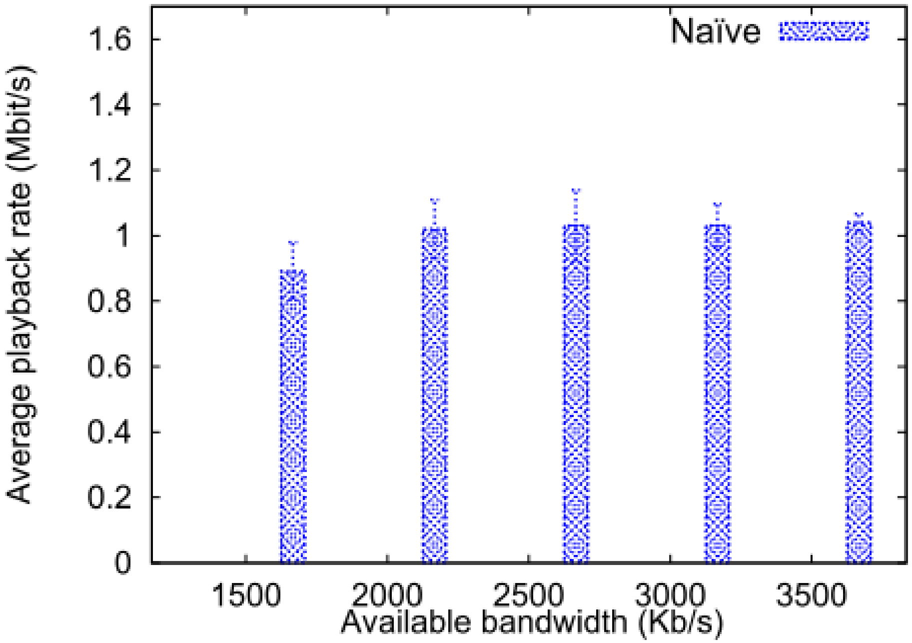
- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3

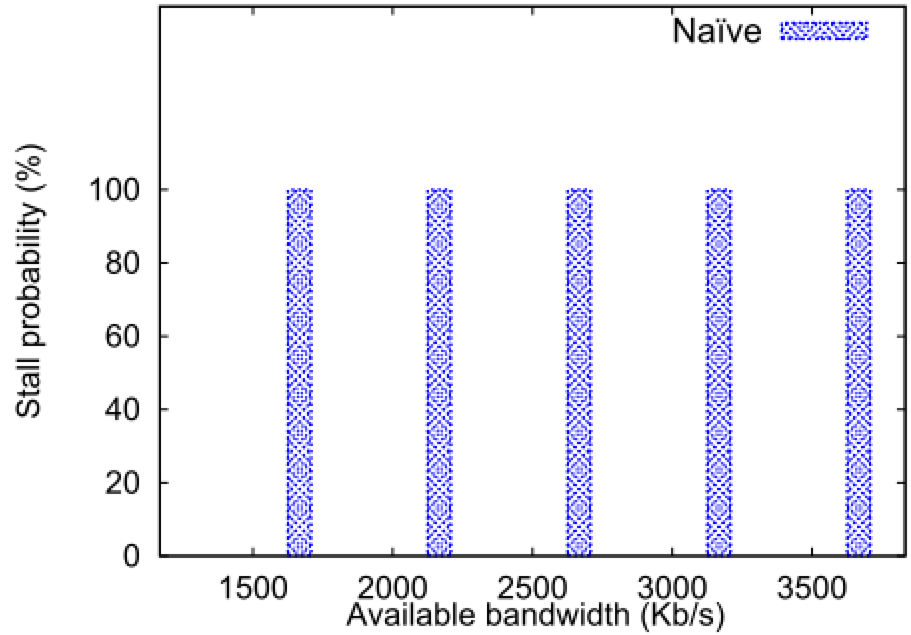
Test Scenario



- Default scenario:
  - Chunks per segment: 5
  - Branches per branch point: 4
  - Branch points: 3
- Results are averages over 30 experiments

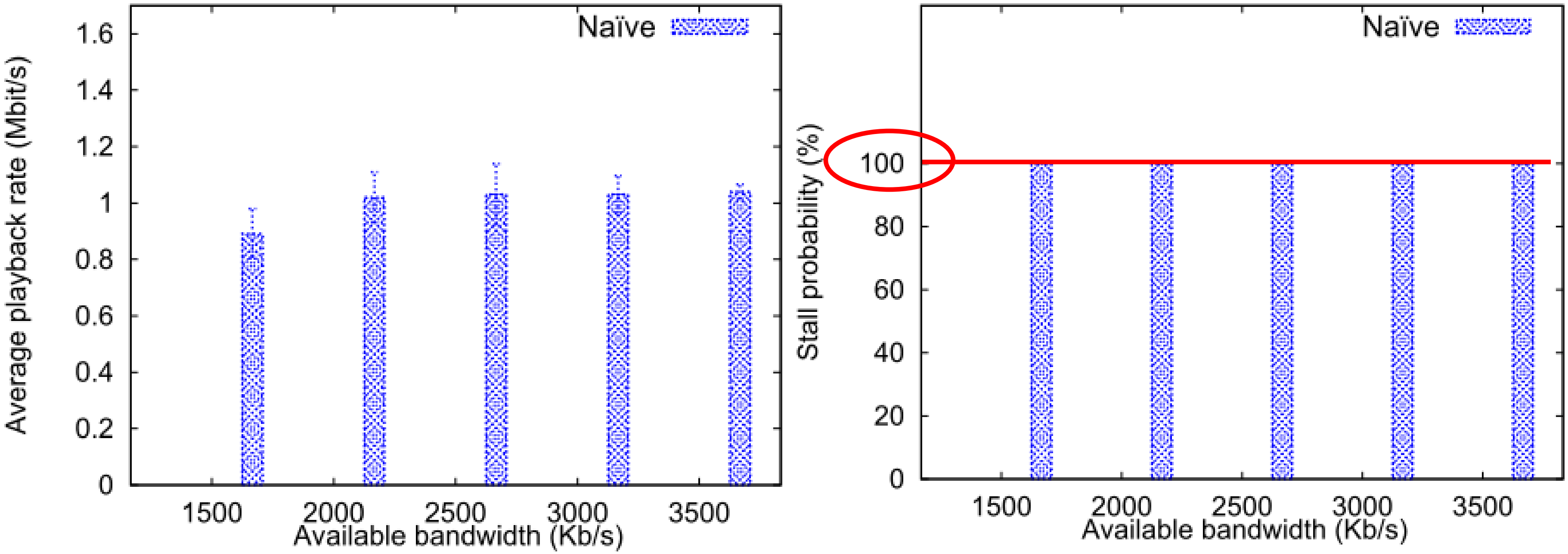
Policy Comparison





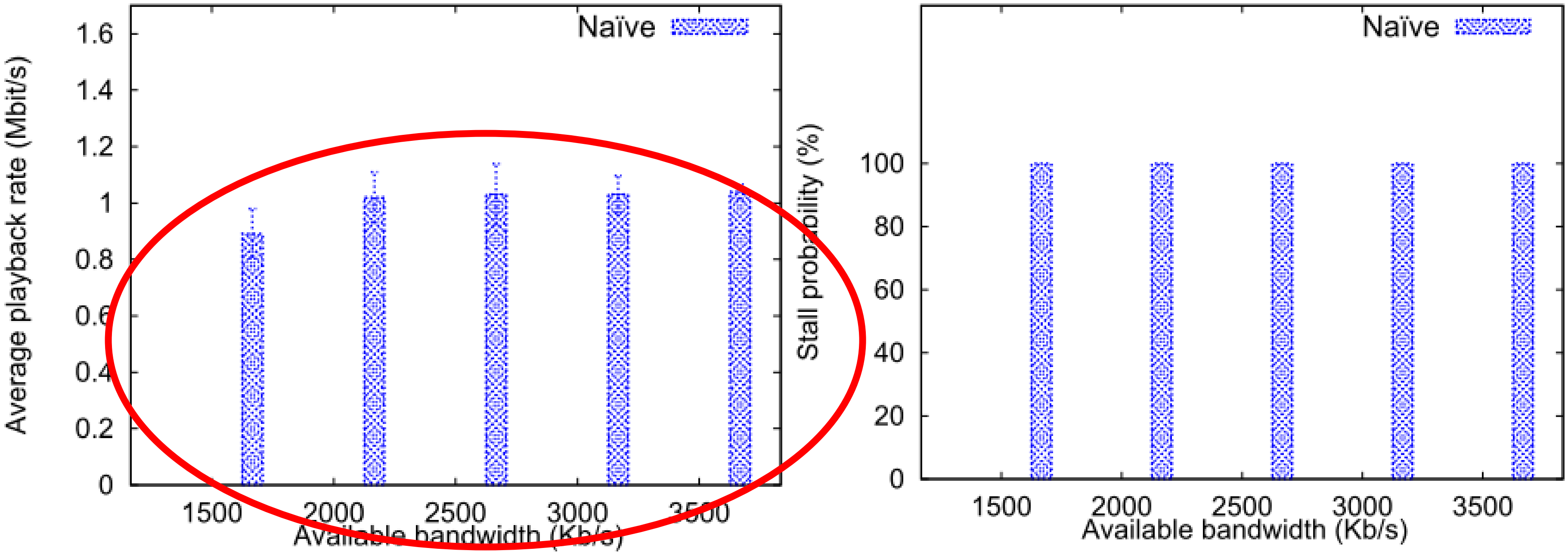
- Naïve policy: does not perform prefetching
  - Stalls at every branch point
  - Note: High playback rate is misleading on its own

Policy Comparison



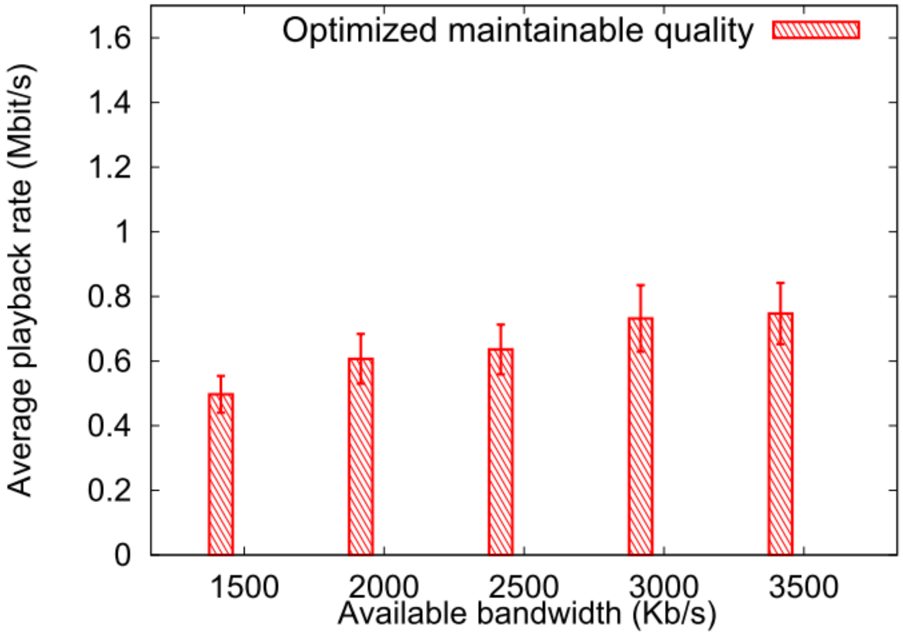
- Naïve policy: does not perform prefetching
  - Stalls at every branch point
  - Note: High playback rate is misleading on its own

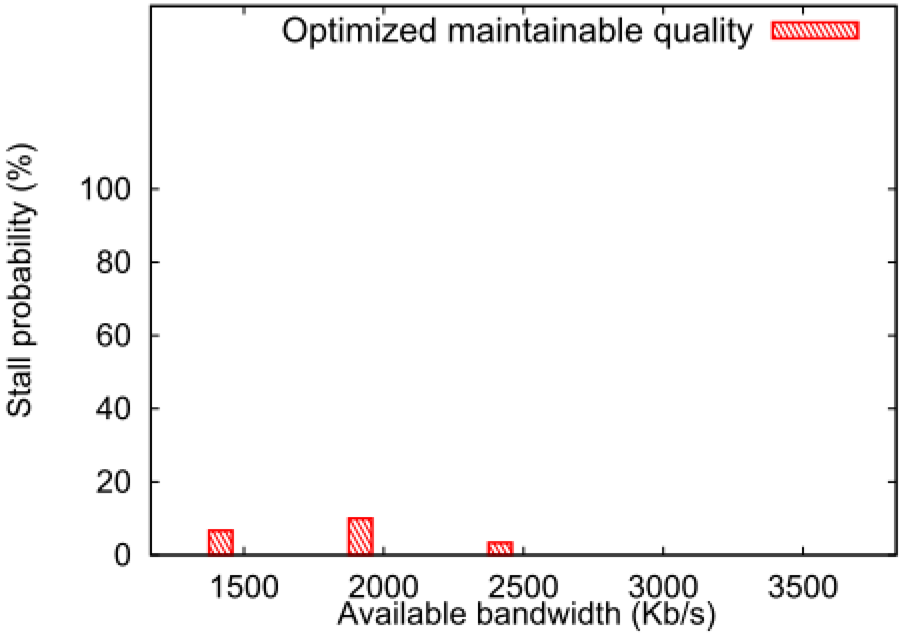
Policy Comparison



- Naïve policy: does not perform prefetching
  - Stalls at every branch point
  - Note: High playback rate is misleading on its own

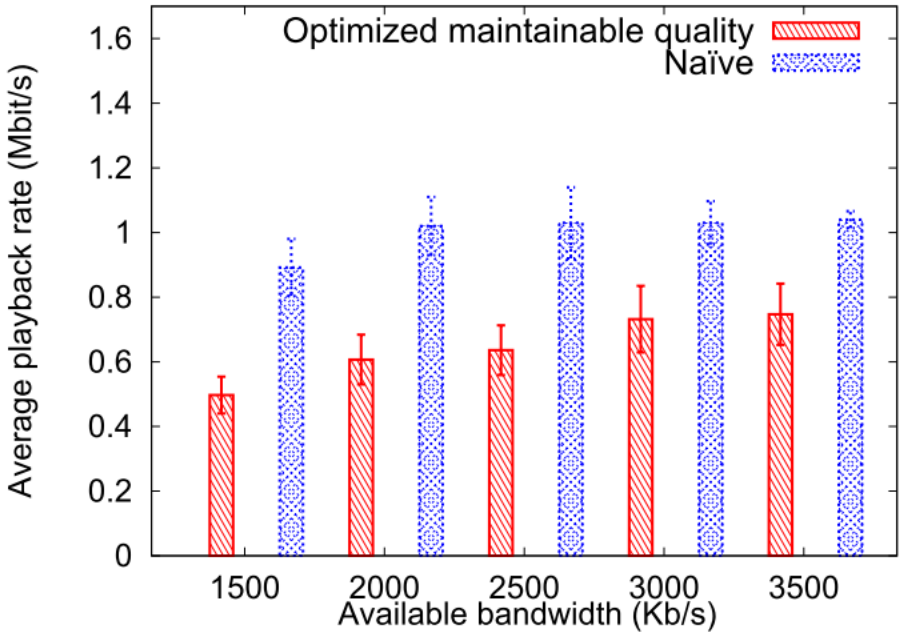
Policy Comparison

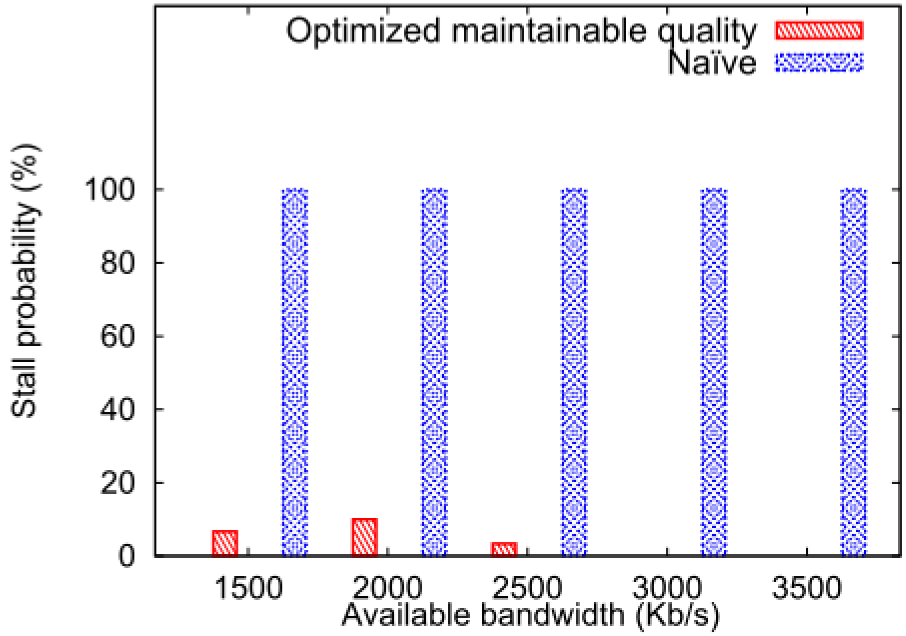




- *Optimized maintainable quality* provides best tradeoff

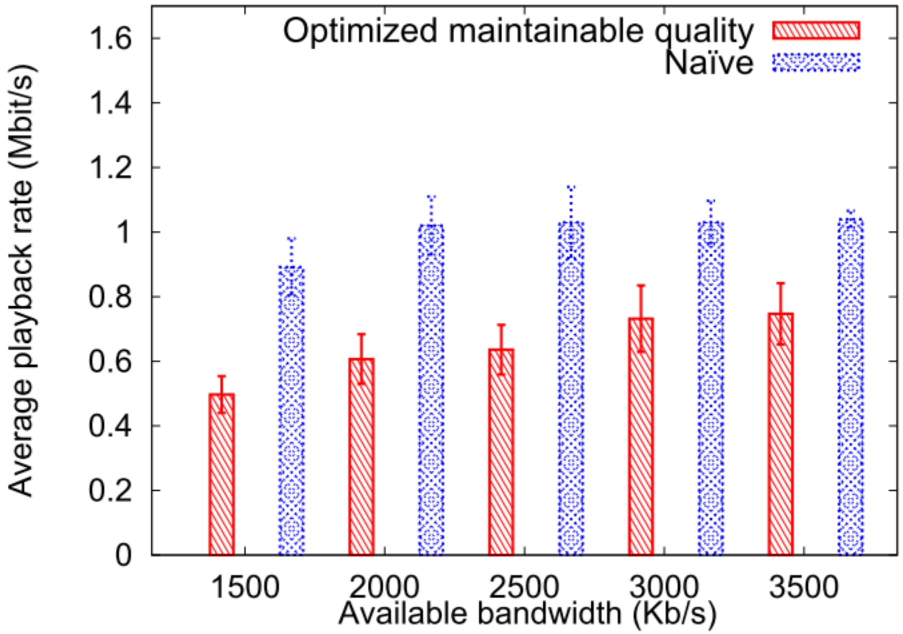
Policy Comparison

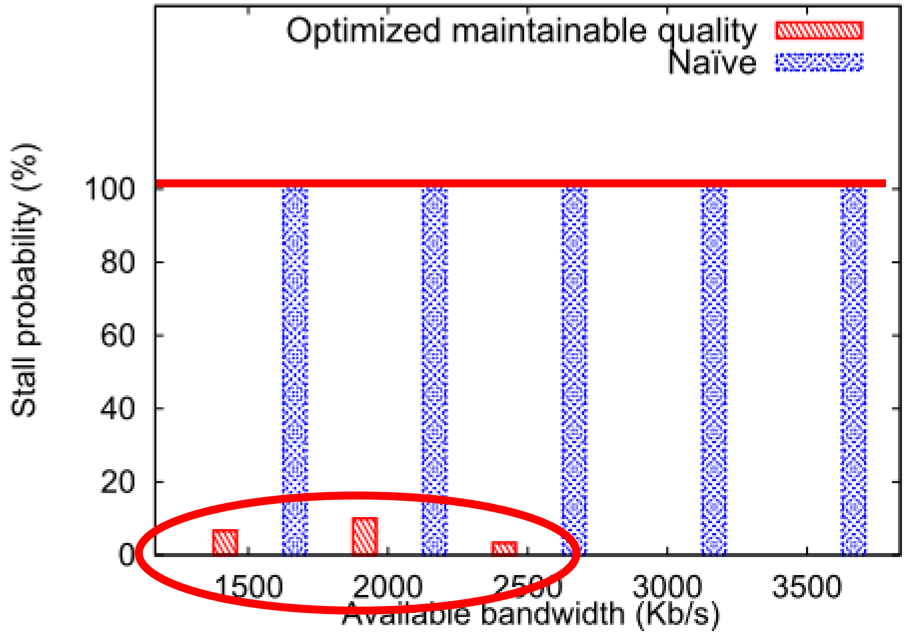




- *Optimized maintainable quality* provides best tradeoff
  - Much lower stall probability
  - Tradeoff is somewhat lower playback rate

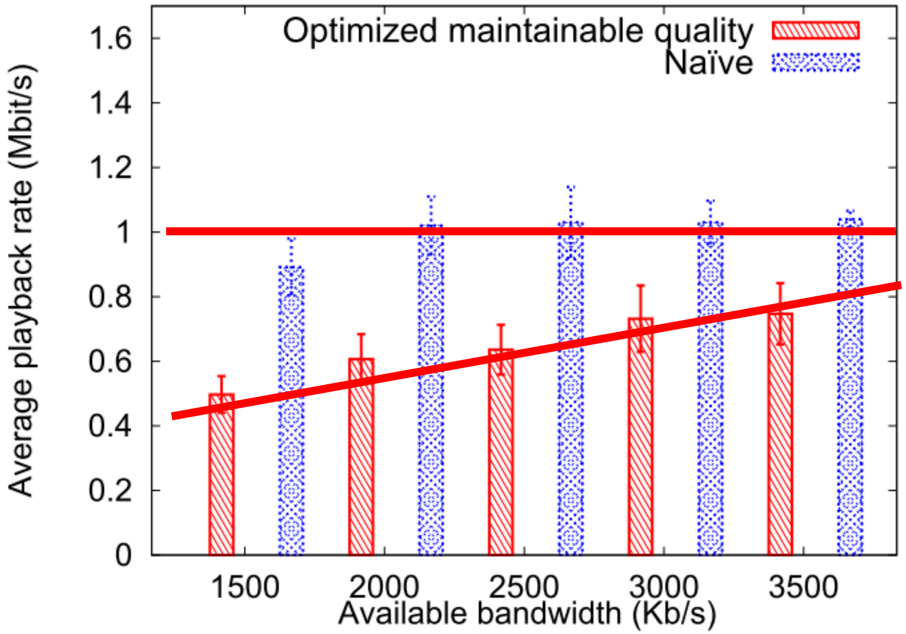
Policy Comparison

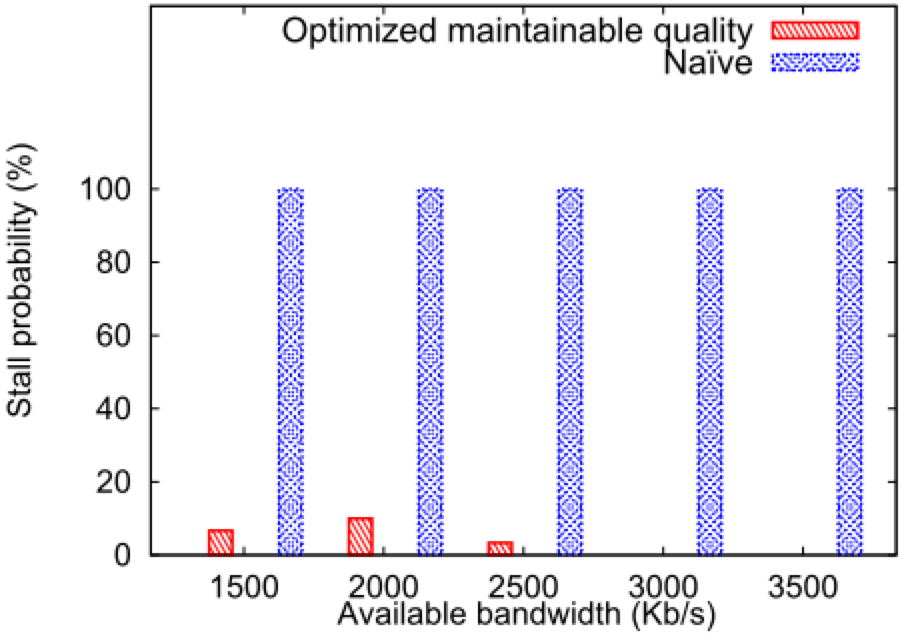




- *Optimized maintainable quality* provides best tradeoff
  - Much lower stall probability
  - Tradeoff is somewhat lower playback rate

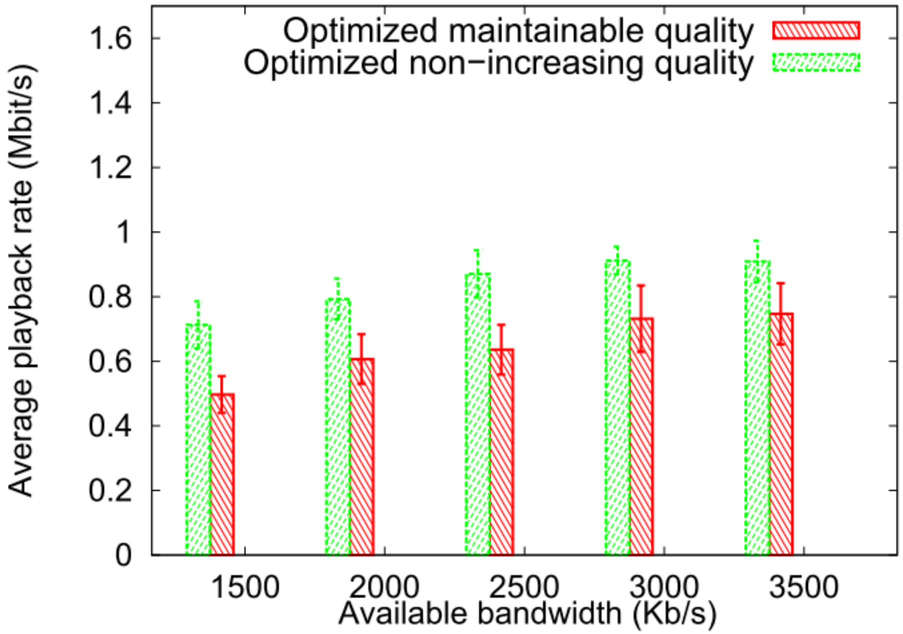
Policy Comparison

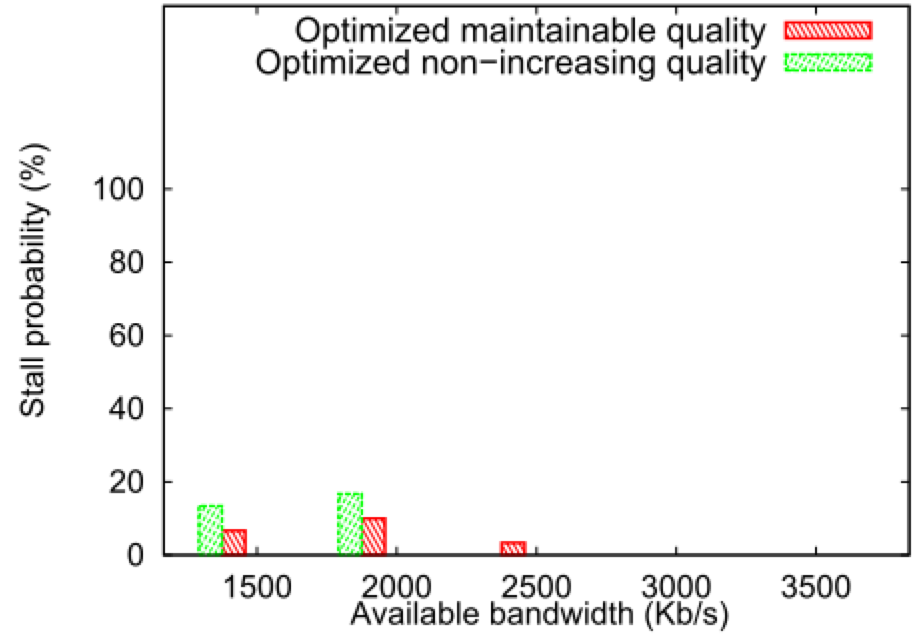




- *Optimized maintainable quality* provides best tradeoff
  - Much lower stall probability
  - Tradeoff is somewhat lower playback rate

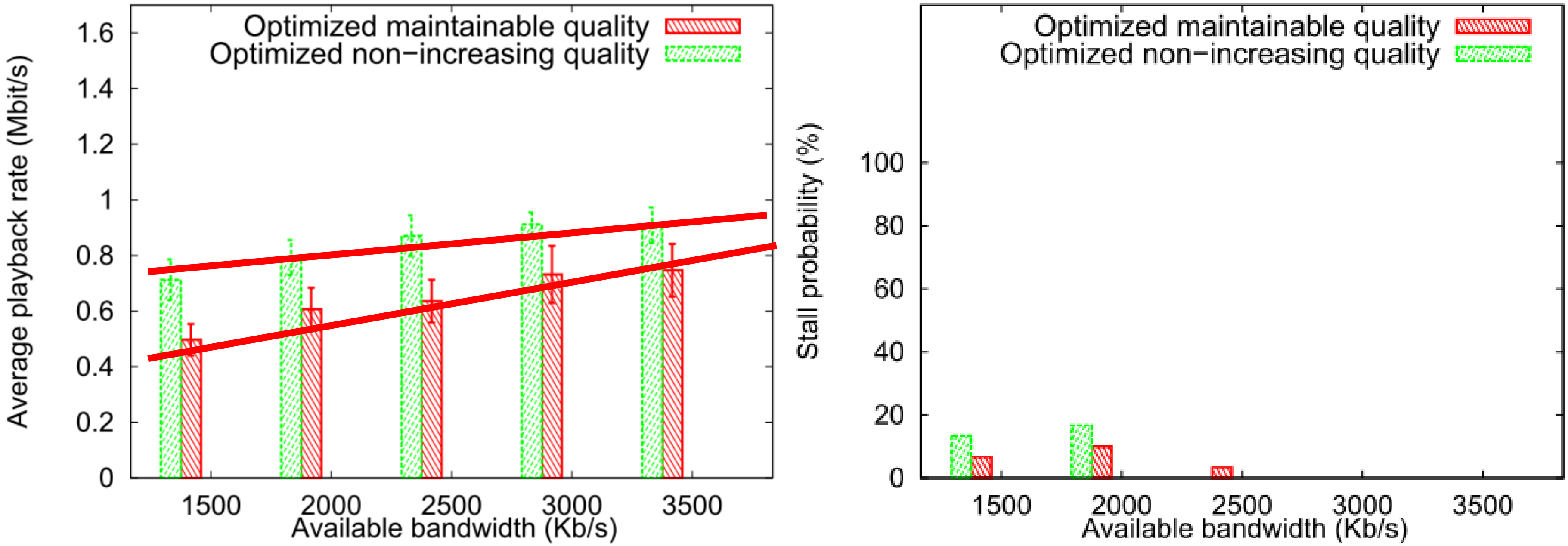
Policy Comparison





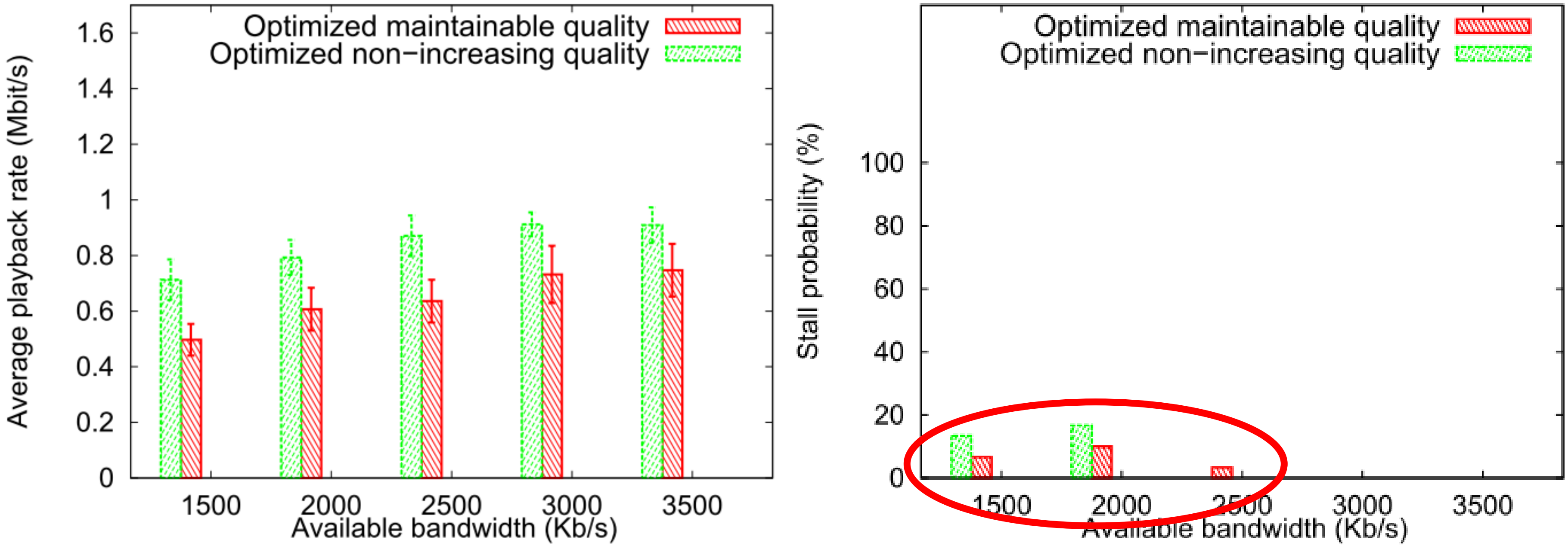
- *Optimized non-increasing quality* is more aggressive
  - Higher playback rate
  - More stalls

Policy Comparison



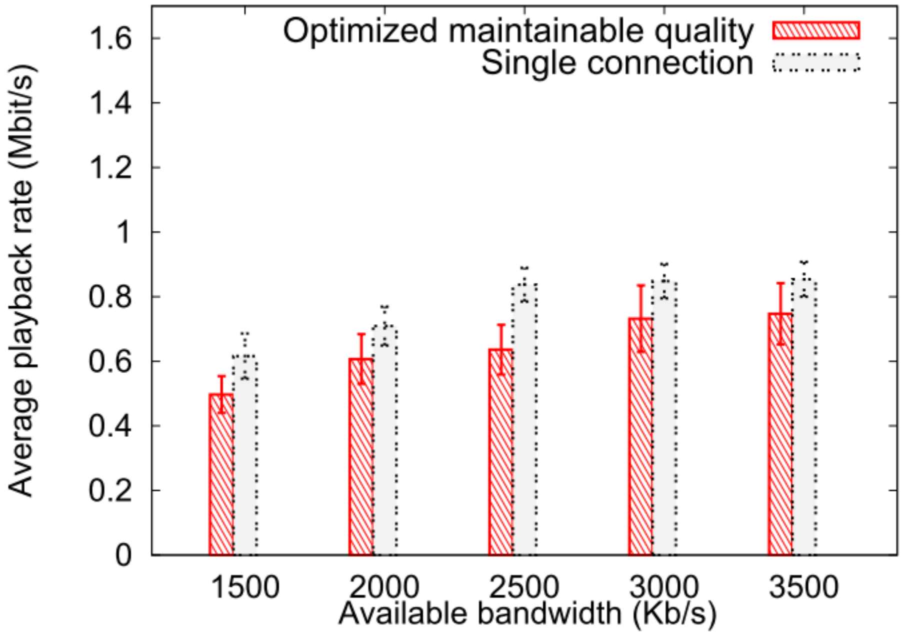
- *Optimized non-increasing quality* is more aggressive
  - Higher playback rate
  - More stalls

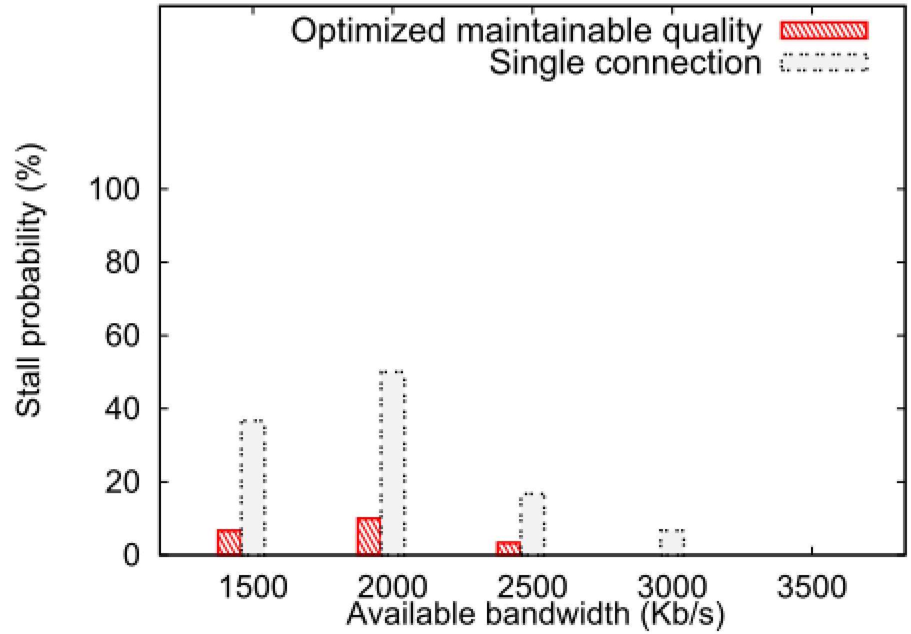
Policy Comparison



- *Optimized non-increasing quality* is more aggressive
  - Higher playback rate
  - More stalls

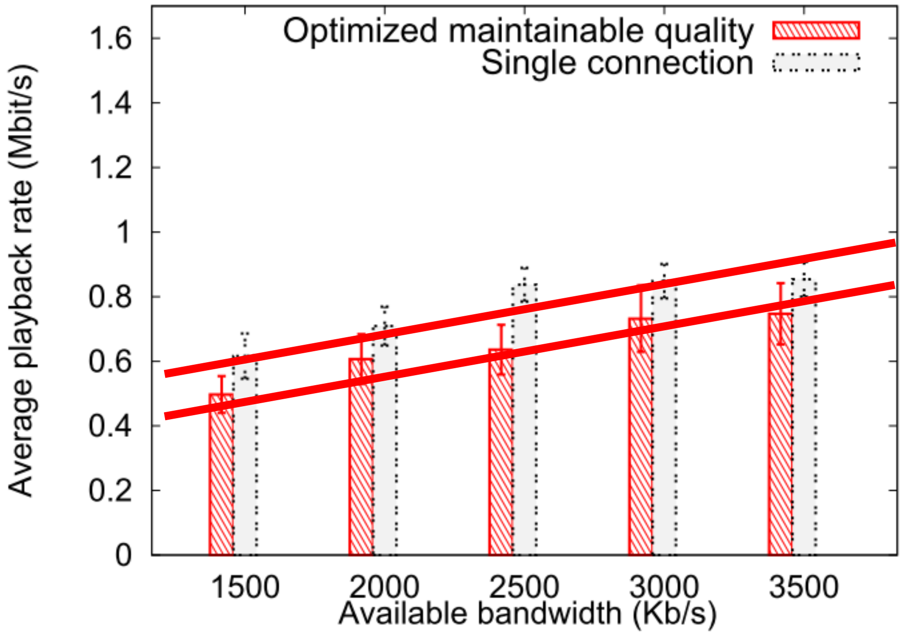
Policy Comparison

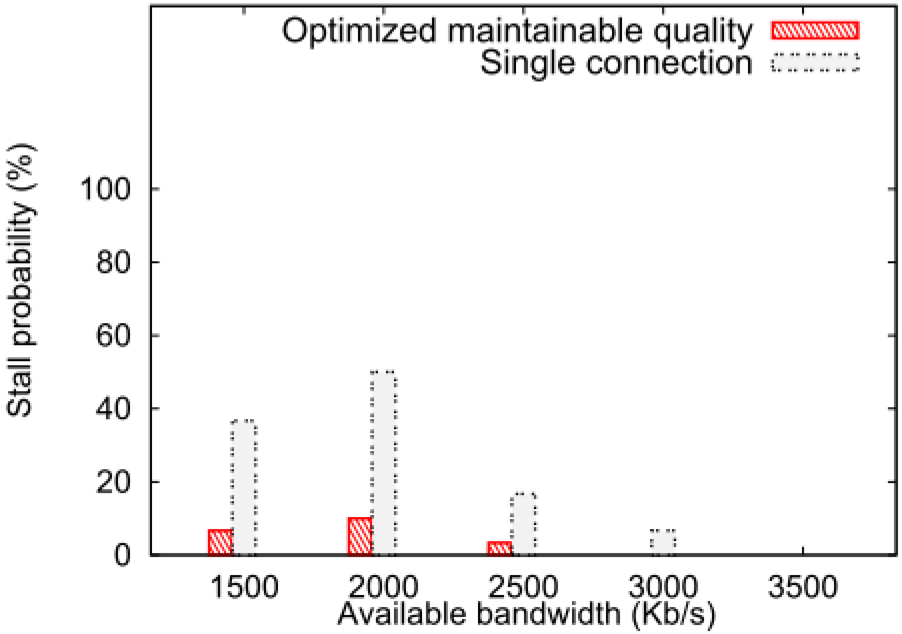




- *Single connection* does not use parallel connections
  - Good (slightly higher) playback rate
  - Much more stalls

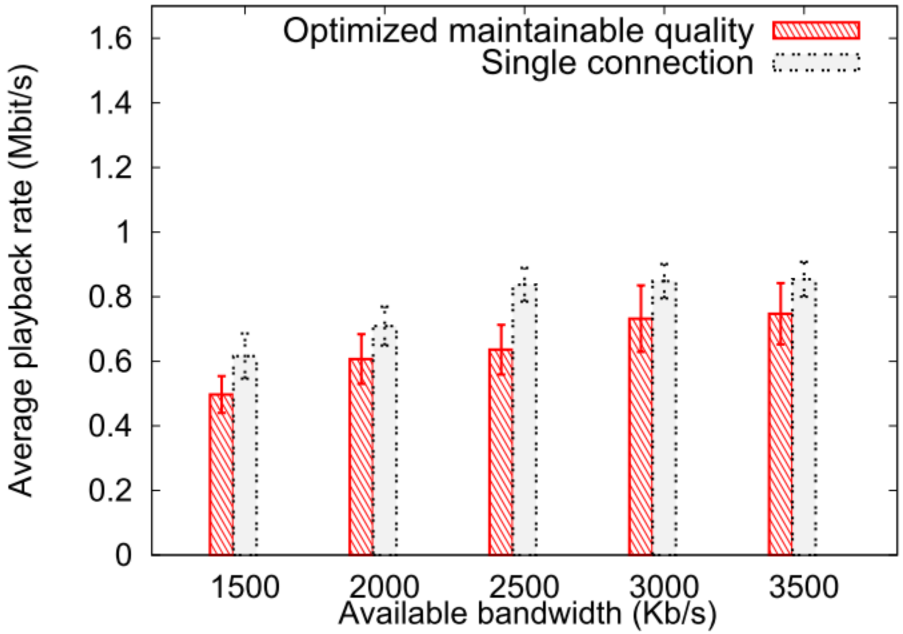
Policy Comparison

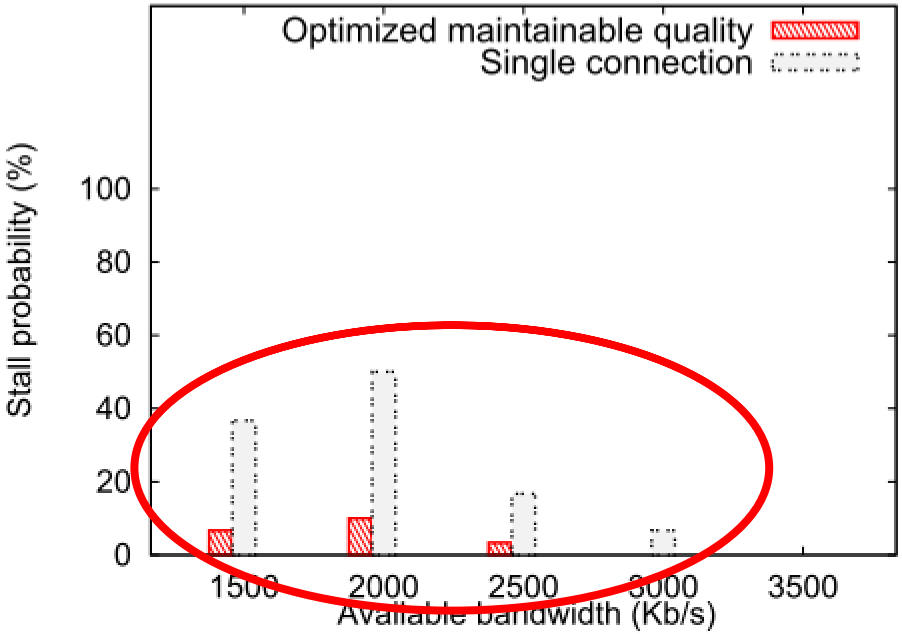




- *Single connection* does not use parallel connections
  - Good (slightly higher) playback rate
  - Much more stalls

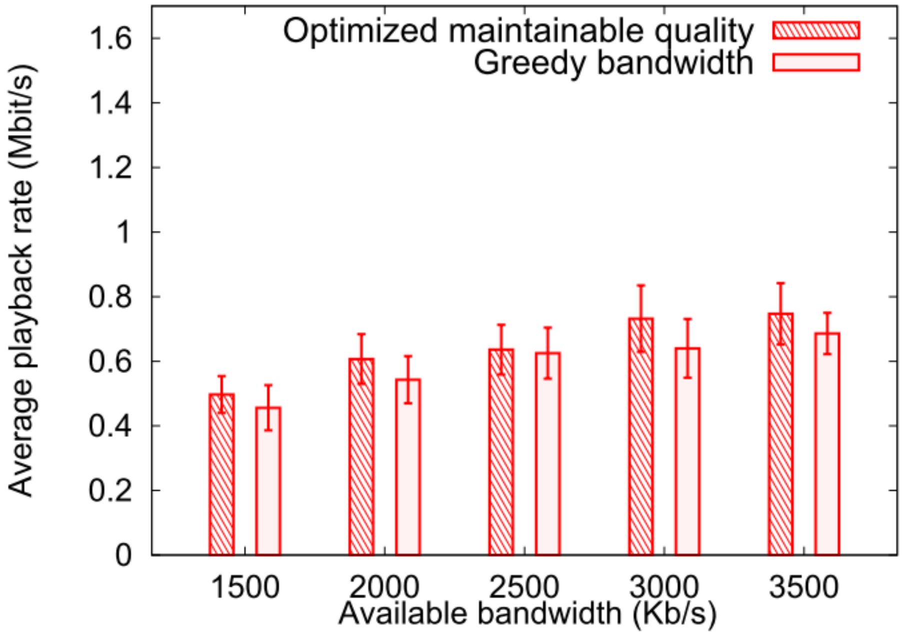
Policy Comparison

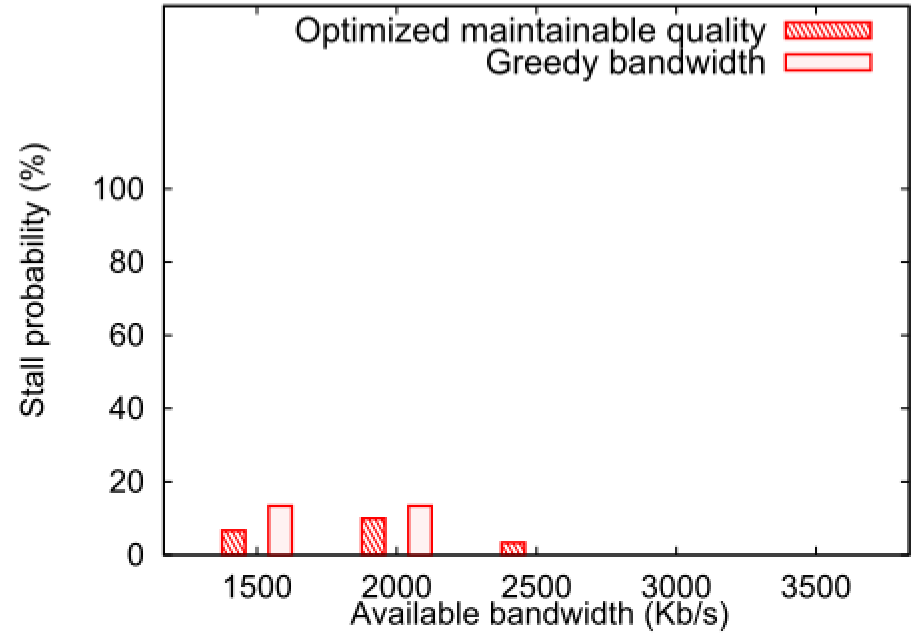




- *Single connection* does not use parallel connections
  - Good (slightly higher) playback rate
  - Much more stalls

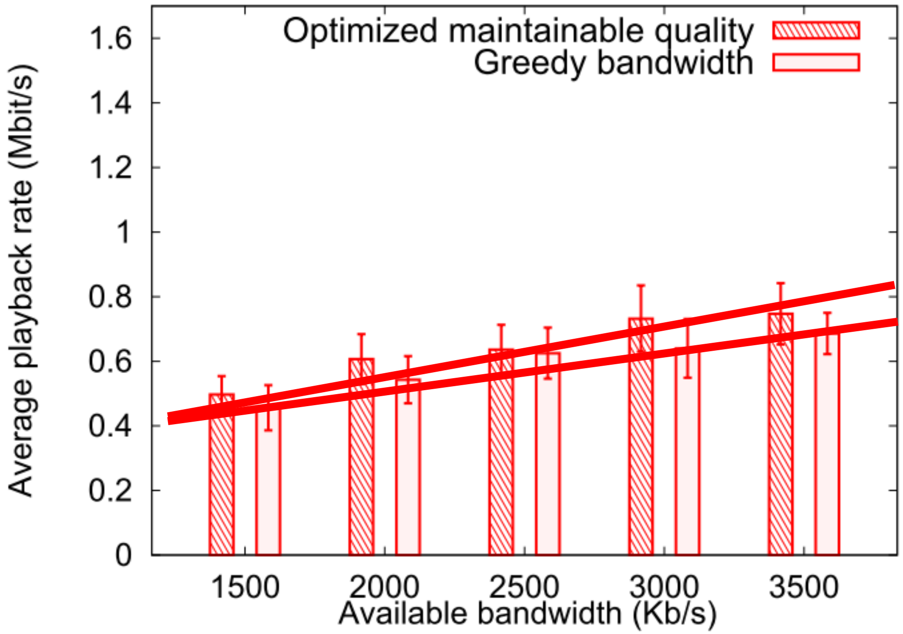
Policy Comparison

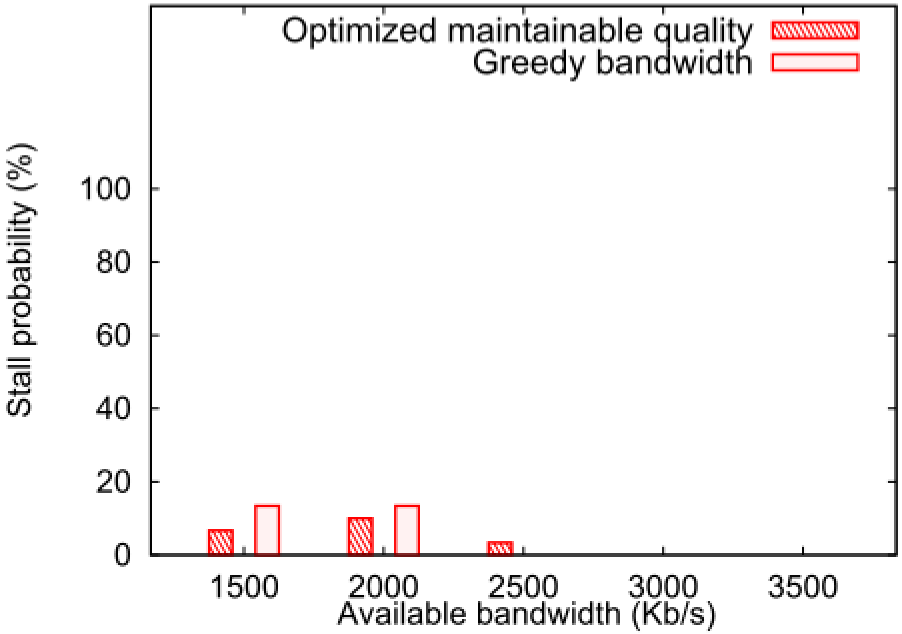




- *Greedy bandwidth* aggressively grabs bandwidth
  – Lower playback rate
  – More stalls

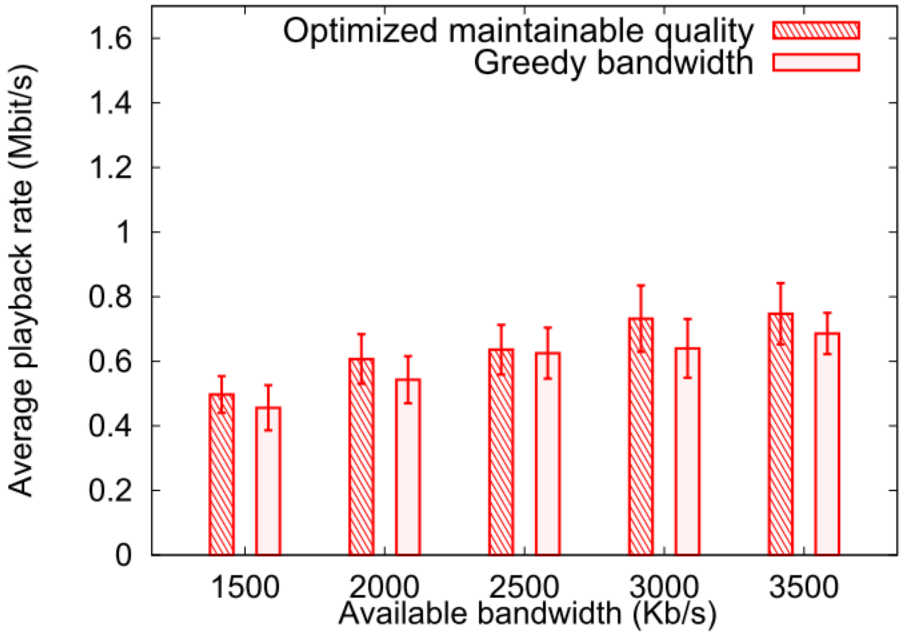
Policy Comparison

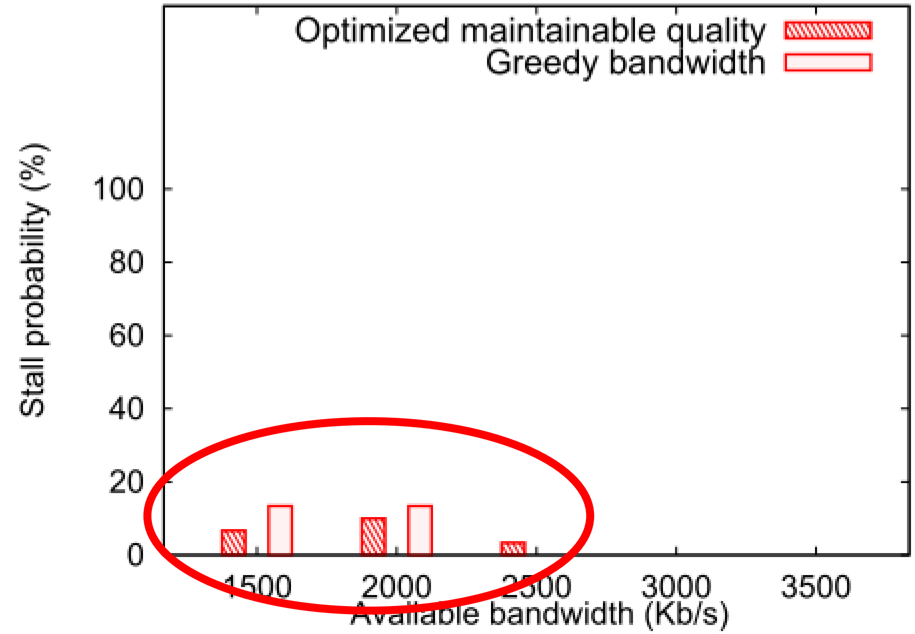




- *Greedy bandwidth* aggressively grabs bandwidth
  - Lower playback rate
  - More stalls

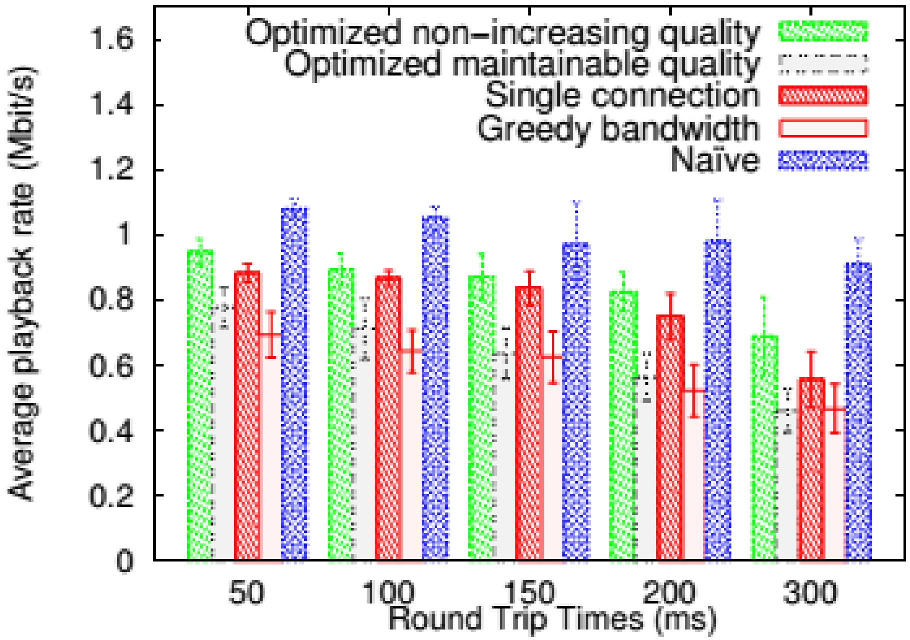
Policy Comparison

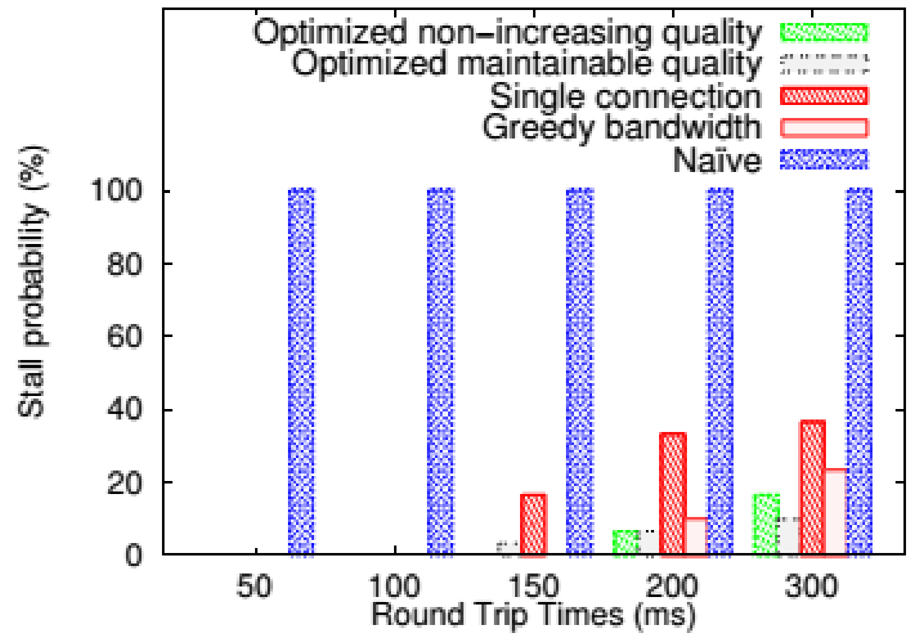




- *Greedy bandwidth* aggressively grabs bandwidth
  - Lower playback rate
  - More stalls

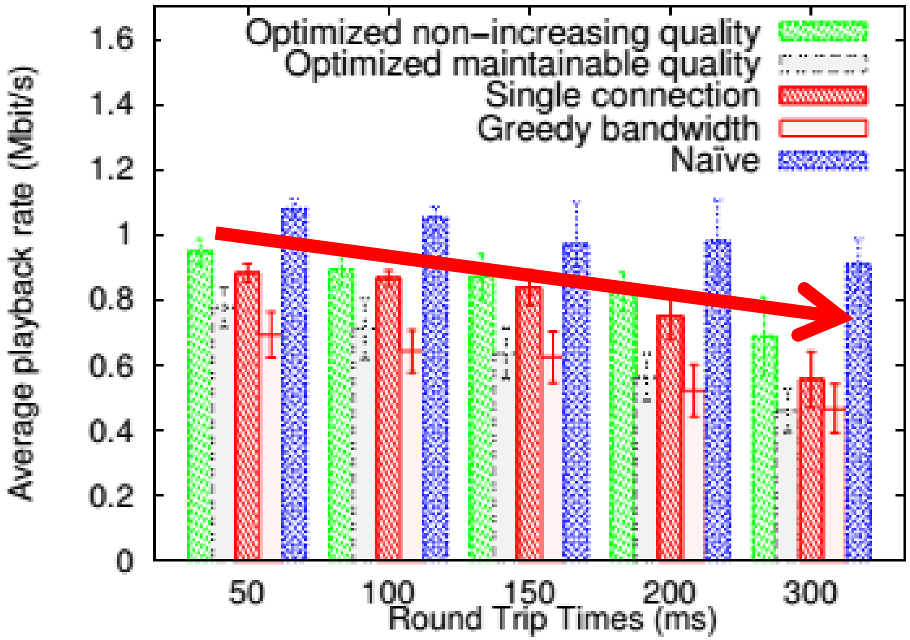
Impact of Round-trip Times (RTTs)

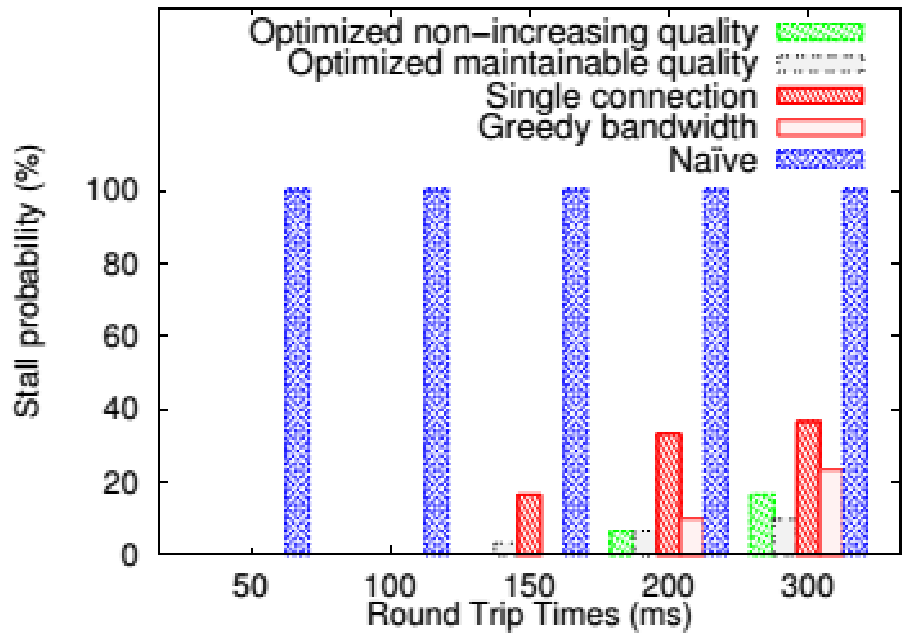




- Quality decreases with larger RTTs
  – Playback rate decrease with RTT
  – Stall probability increase with RTT

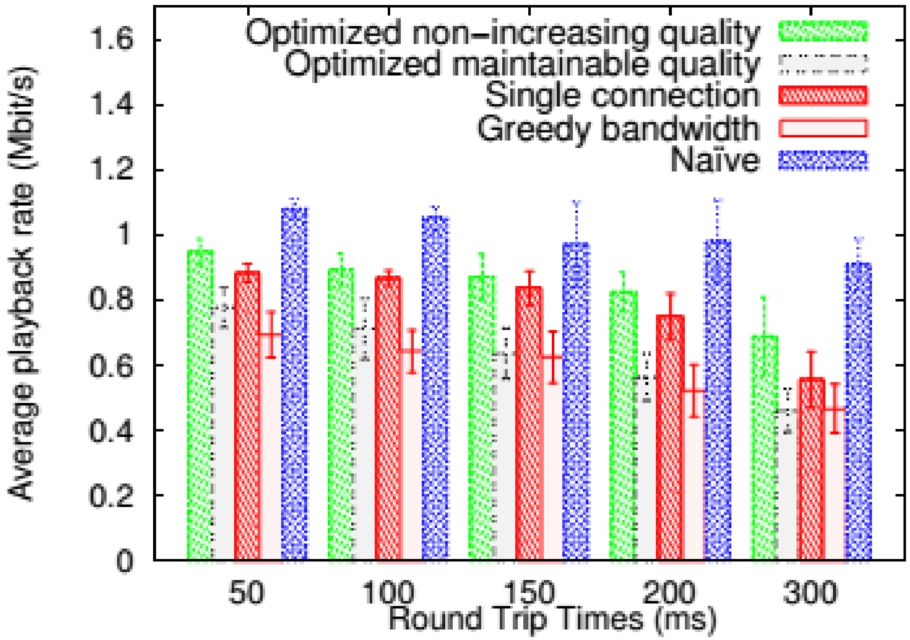
Impact of Round-trip Times (RTTs)

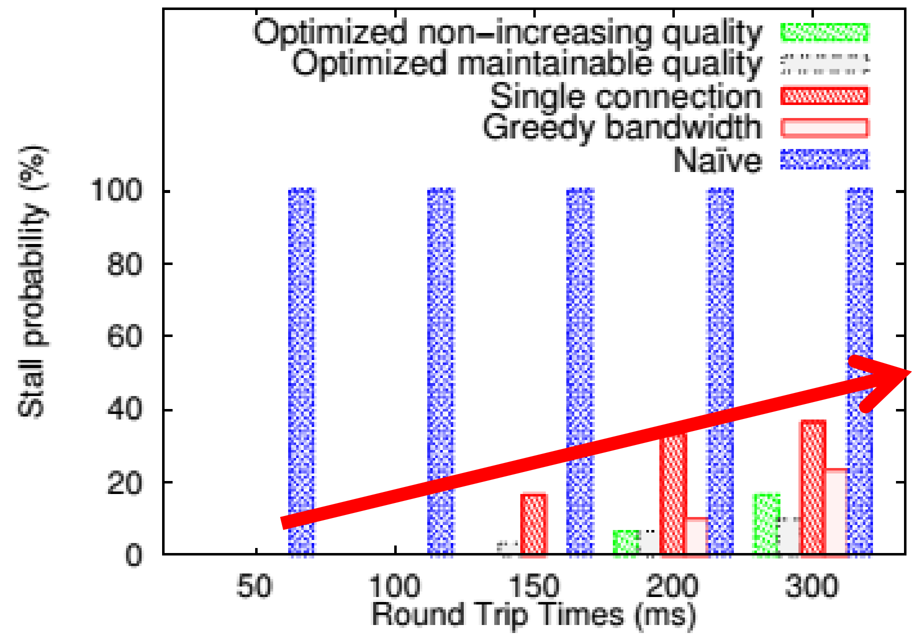




- Quality decreases with larger RTTs
  - Playback rate decrease with RTT
  - Stall probability increase with RTT

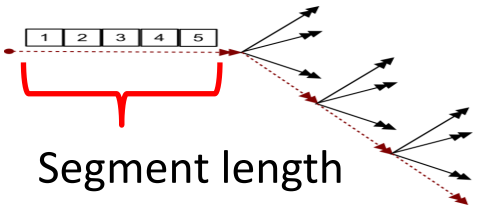
Impact of Round-trip Times (RTTs)



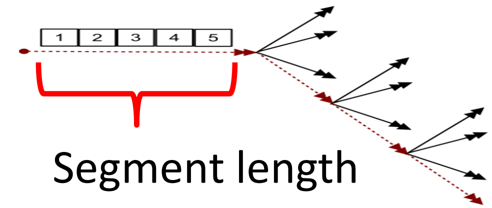


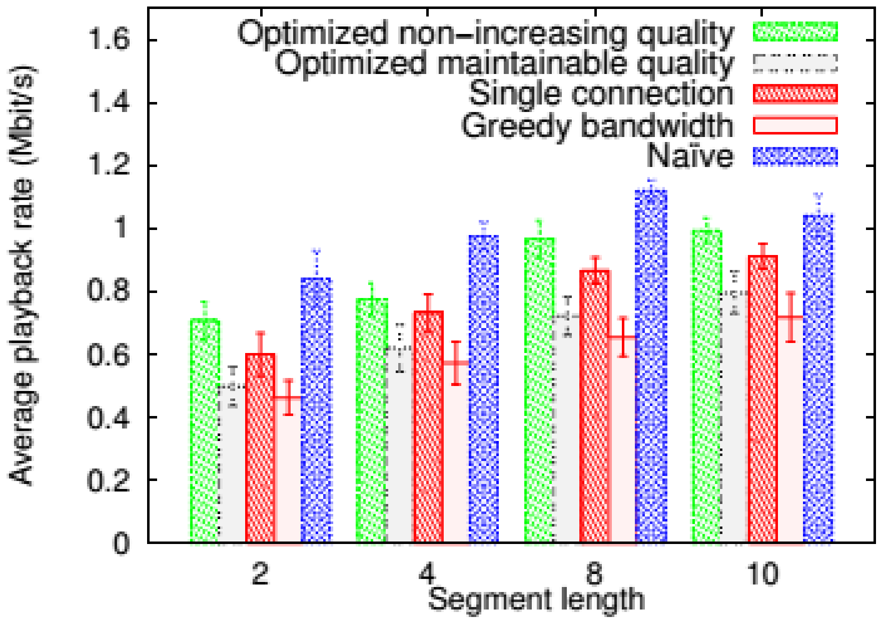
- Quality decreases with larger RTTs
  - Playback rate decrease with RTT
  - Stall probability increase with RTT

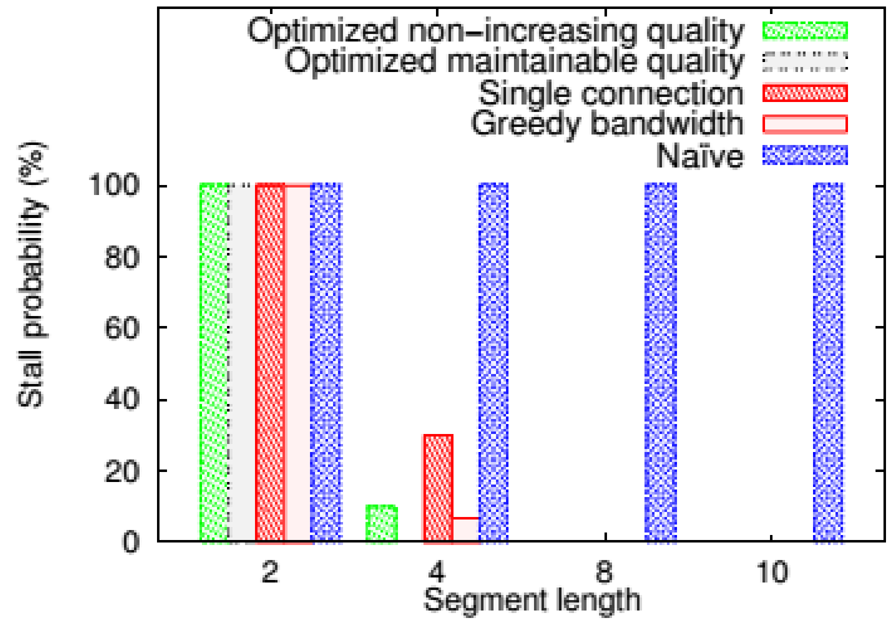
Impact of Segment Lengths



Impact of Segment Lengths

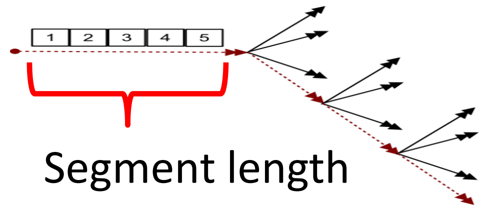


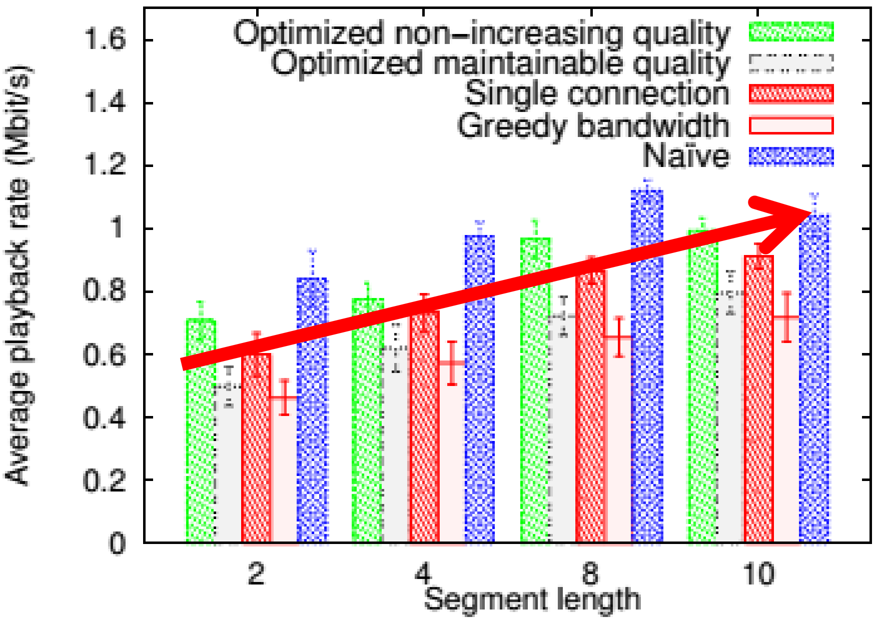


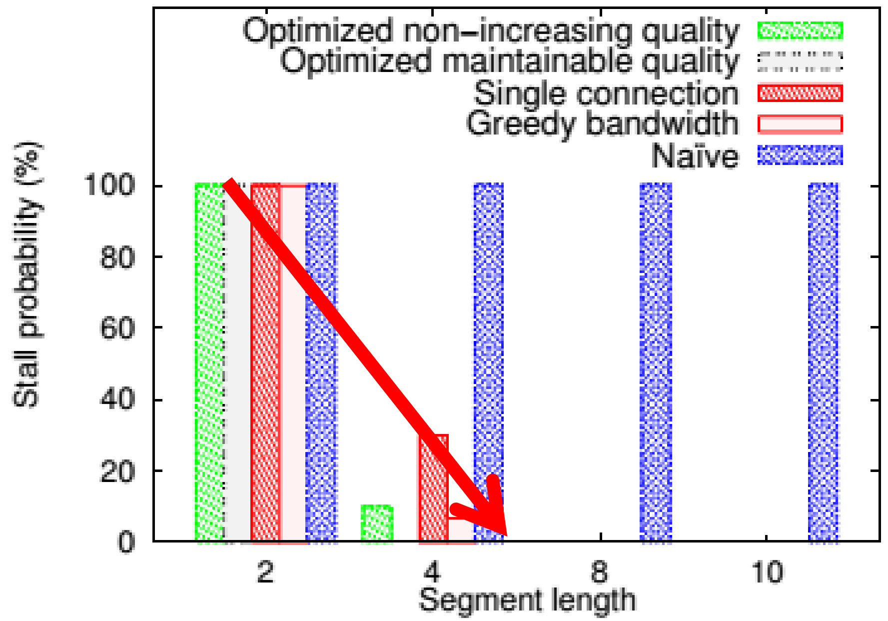


- Quality increases with more chunks per segment
- Very many stalls if segments are too short

Impact of Segment Lengths

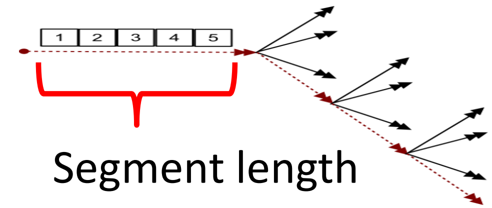


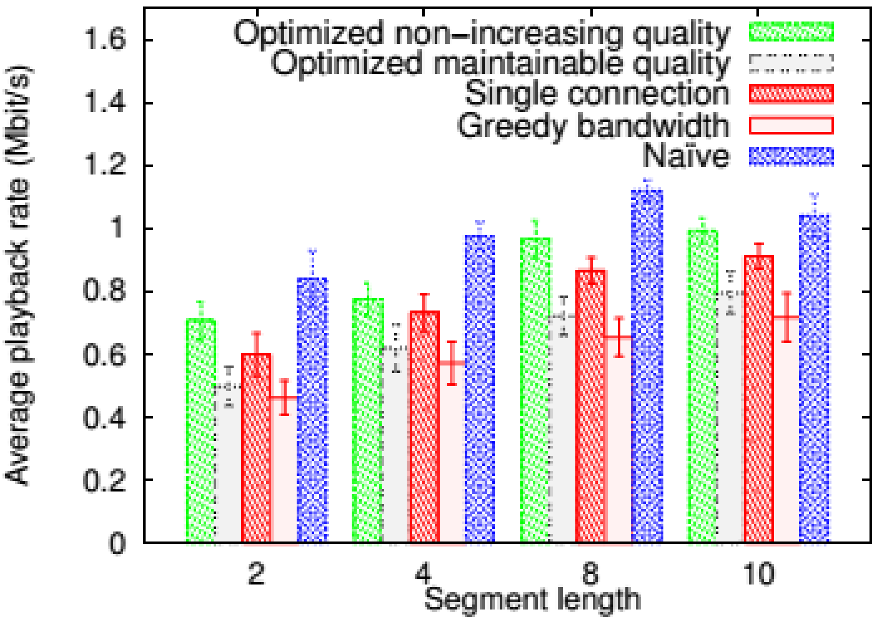


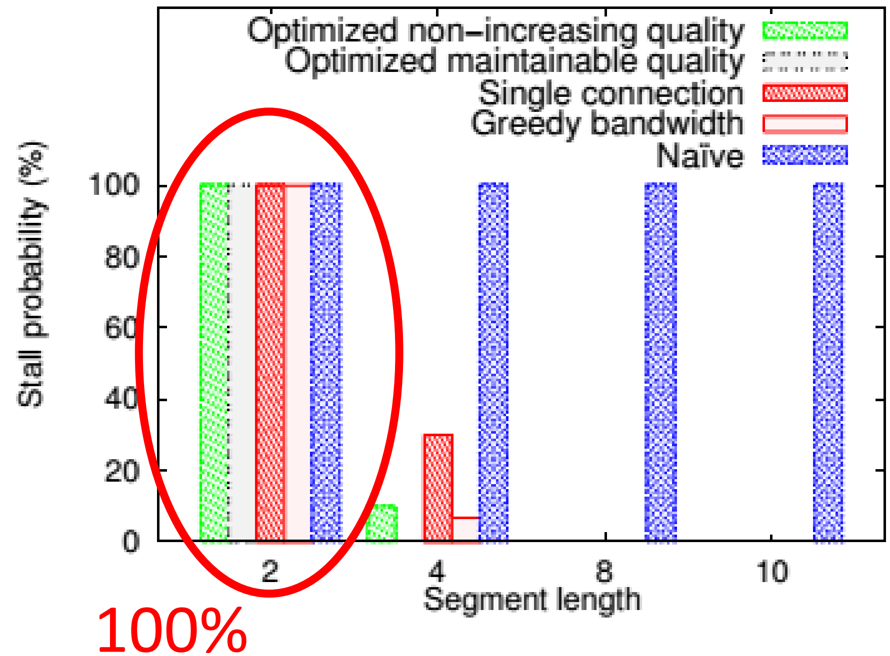


- Quality increases with more chunks per segment
- Very many stalls if segments are too short

Impact of Segment Lengths

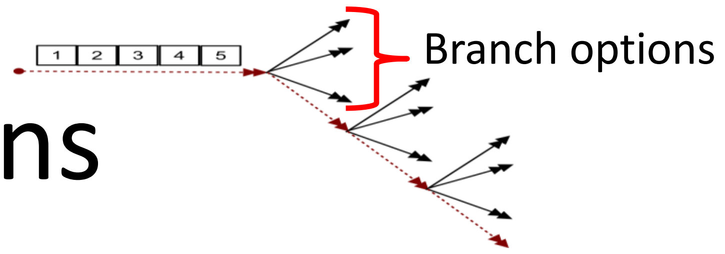


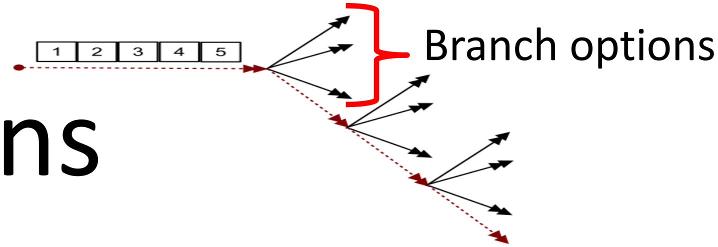




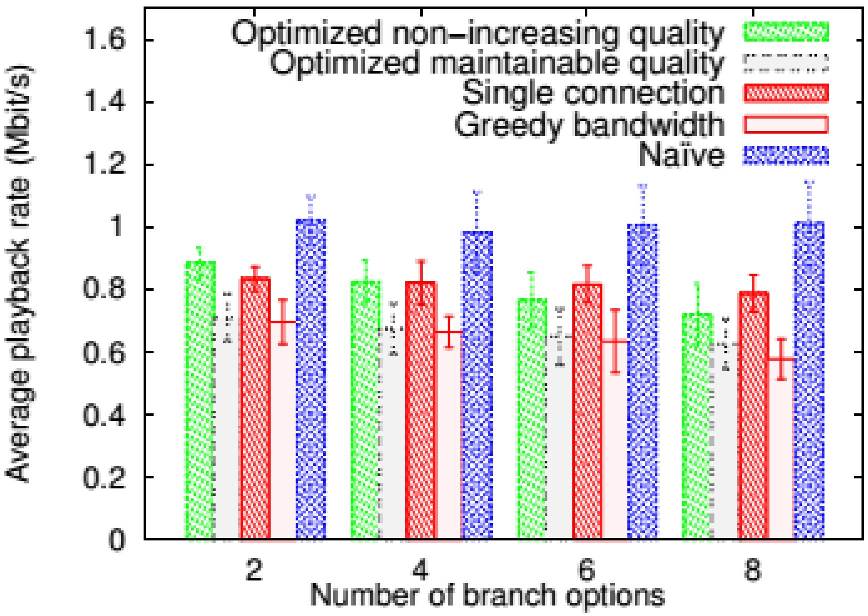
- Quality increases with more chunks per segment
- Very many stalls if segments are too short

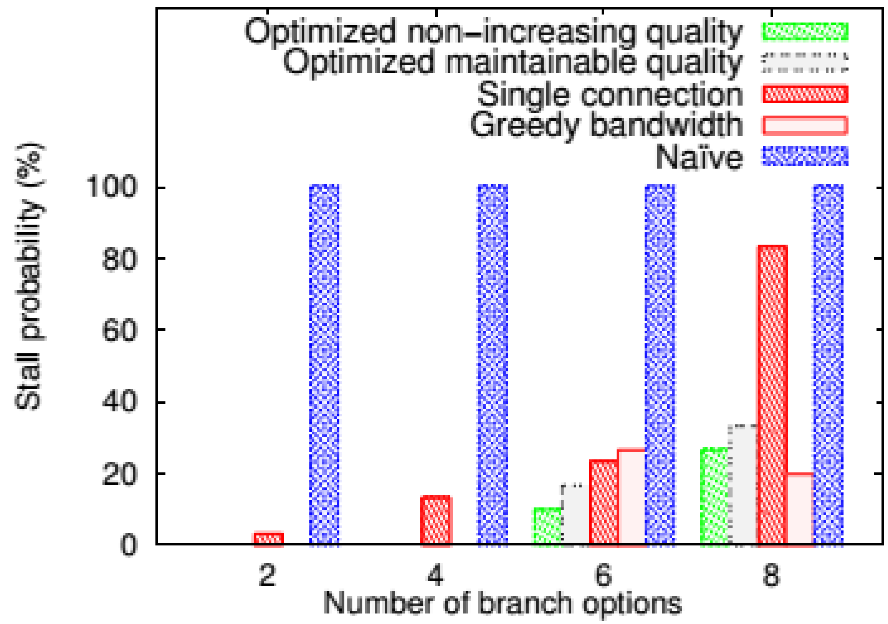
Impact of Branch Options





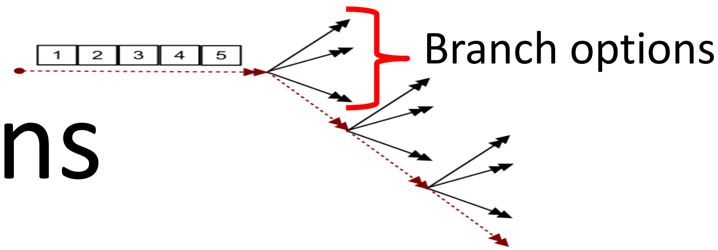
Impact of Branch Options

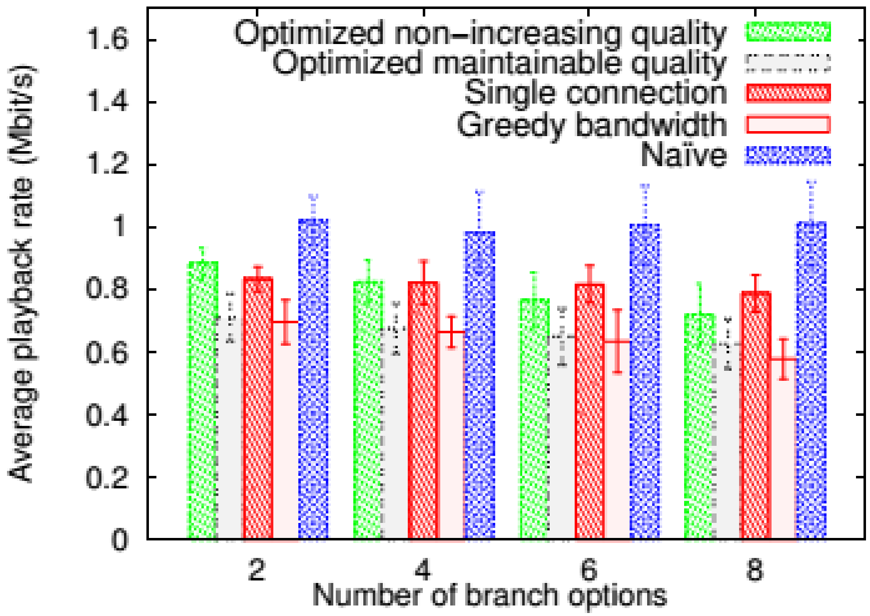


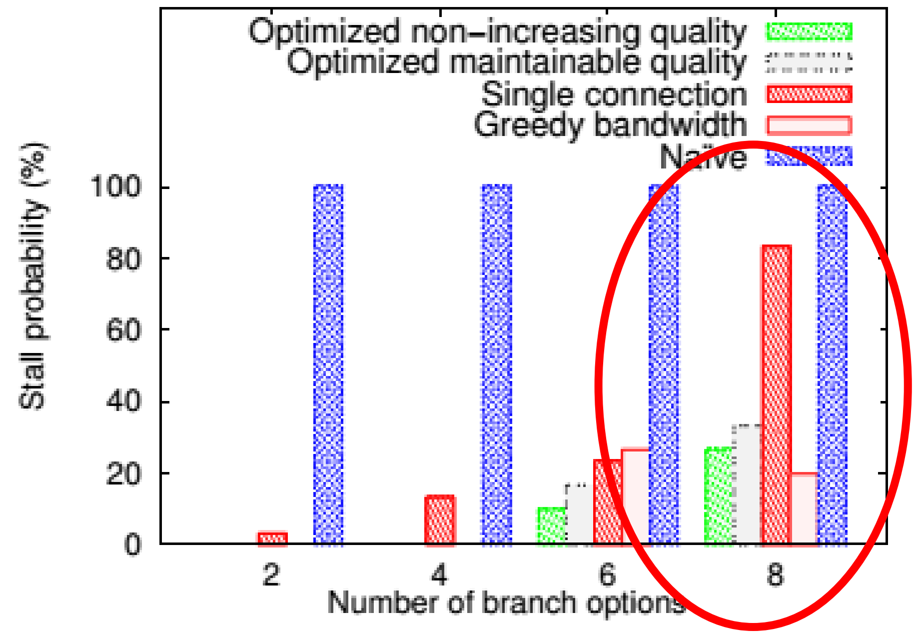


- Stalls frequent when too many branch options
  - Single connection struggles the most

Impact of Branch Options

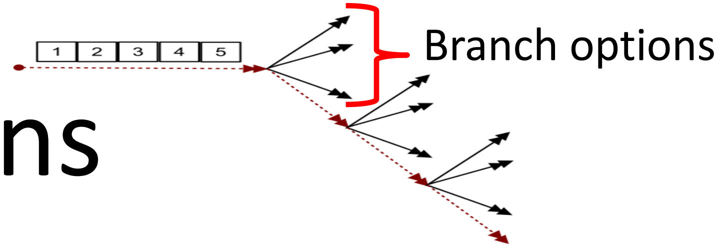


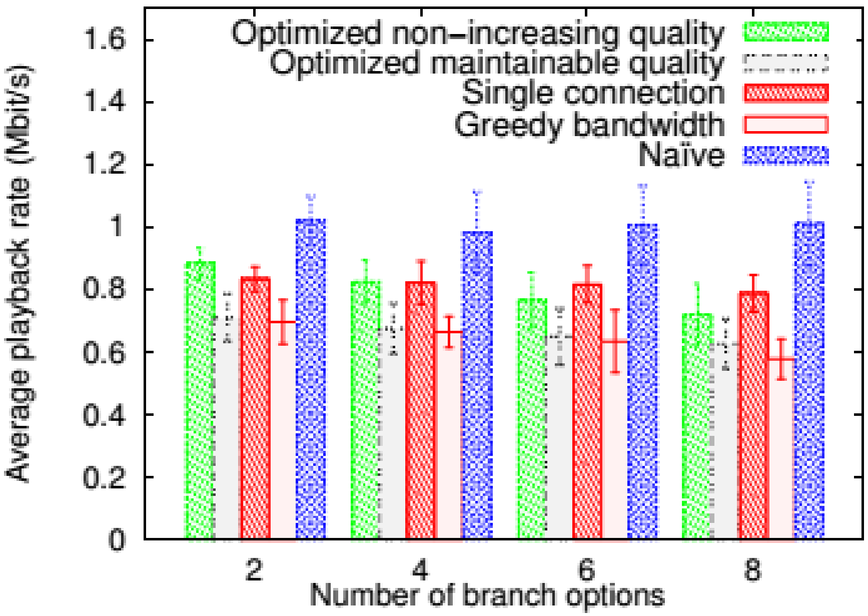


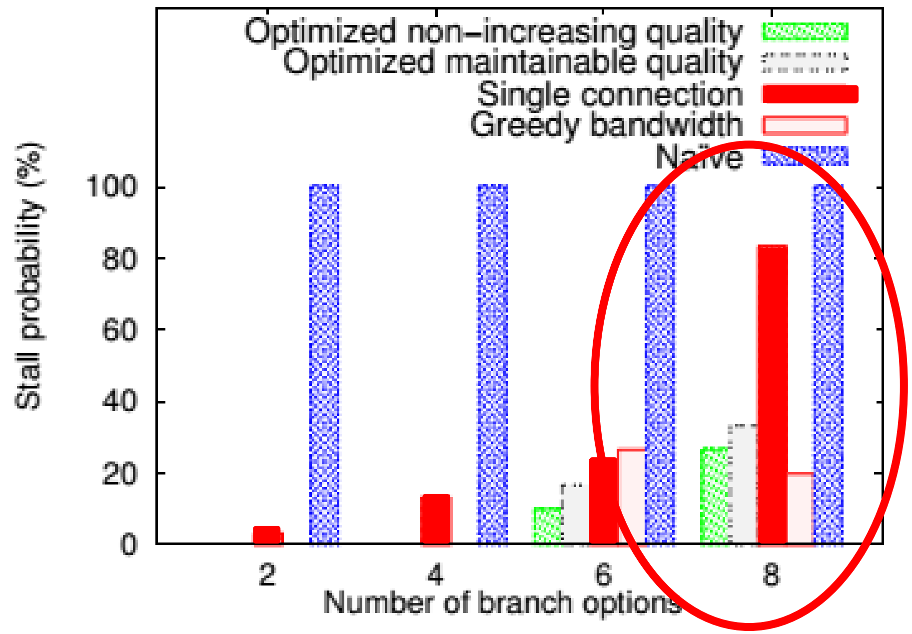


- Stalls frequent when too many branch options
  - Single connection struggles the most

Impact of Branch Options

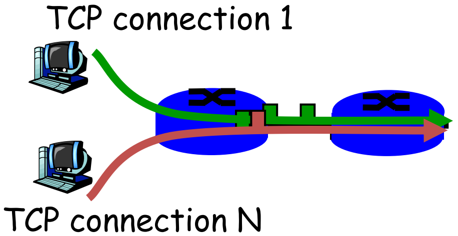




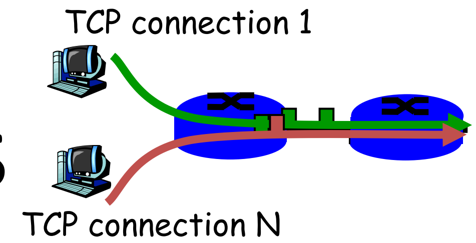


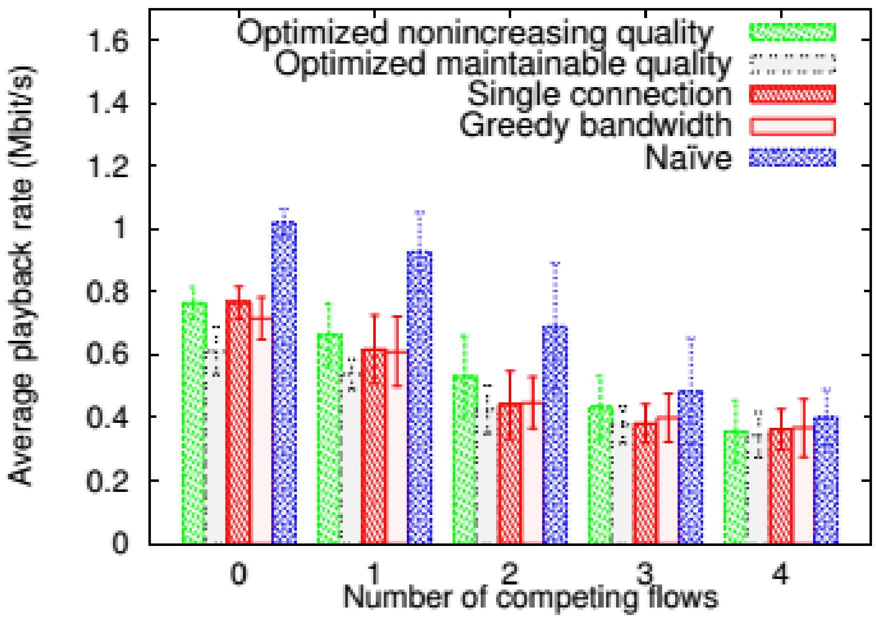
- Stalls frequent when too many branch options
  - Single connection struggles the most

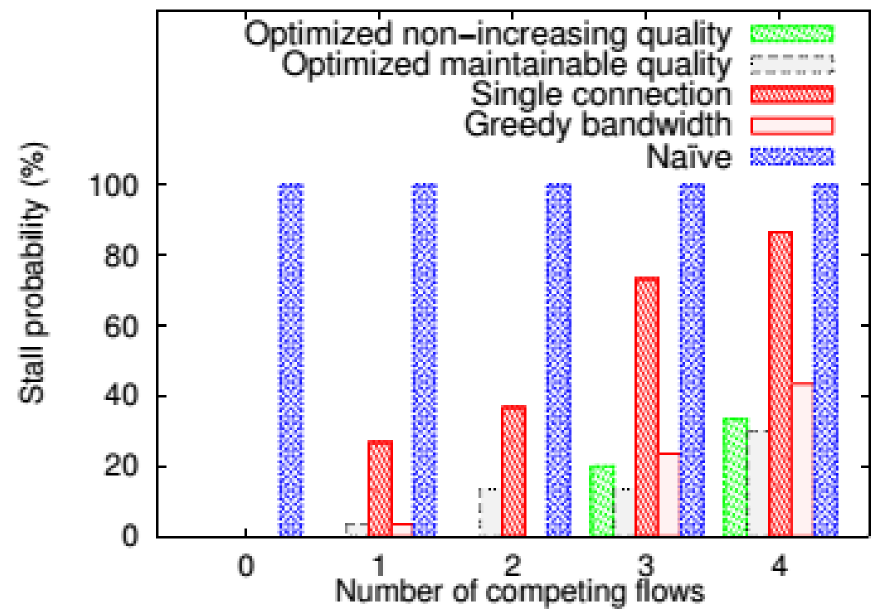
Impact of Competing Flows



Impact of Competing Flows

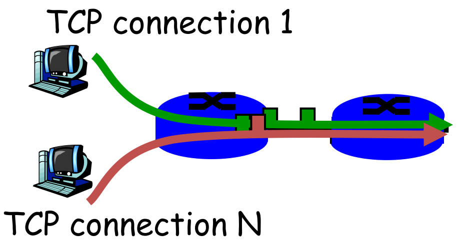


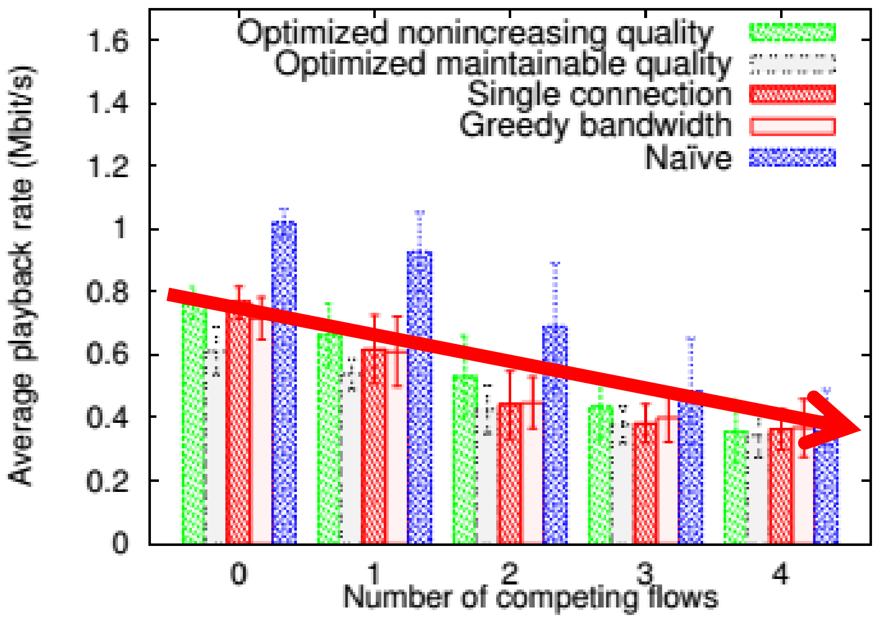


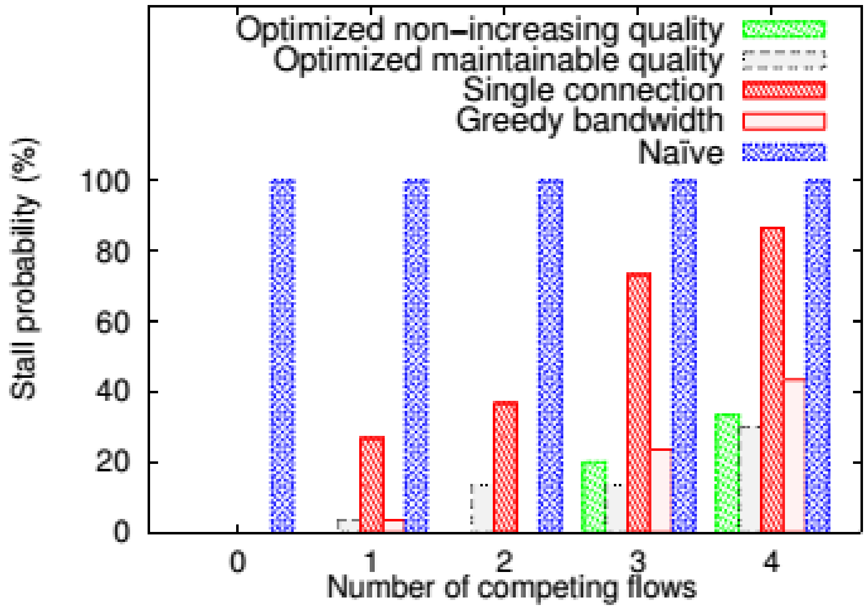


- Player adapts playback rate based on competing traffic
- Parallel connection polices see increased benefits when competing traffic
  - E.g., Single connection policy has much more stalls when competing flows

Impact of Competing Flows

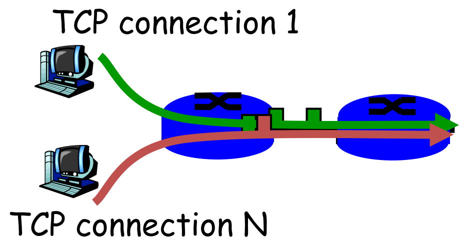


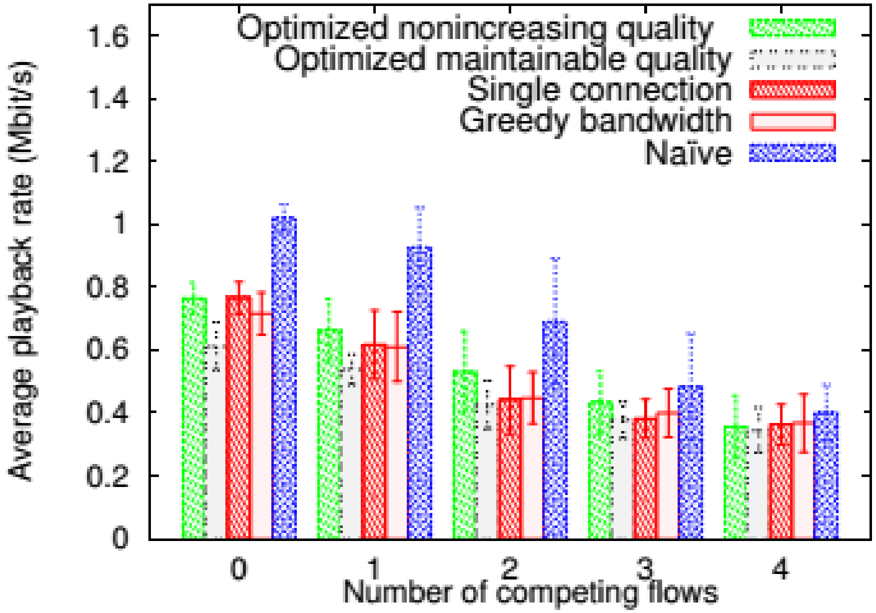


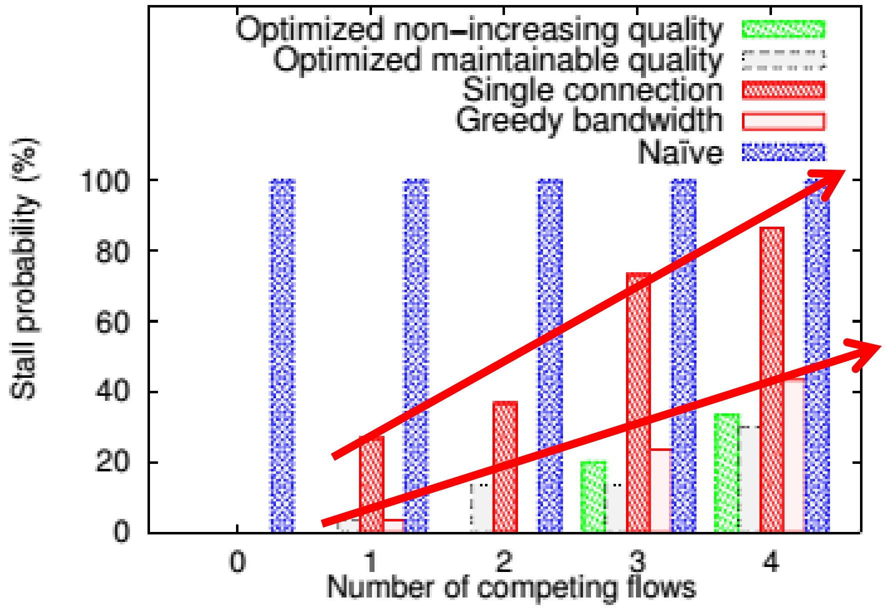


- Player adapts playback rate based on competing traffic
- Parallel connection polices see increased benefits when competing traffic
  - E.g., Single connection policy has much more stalls when competing flows

Impact of Competing Flows

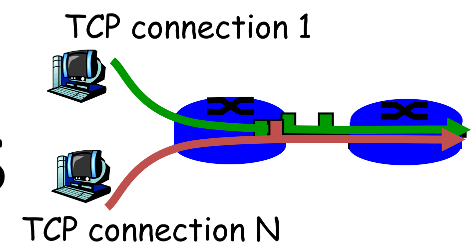


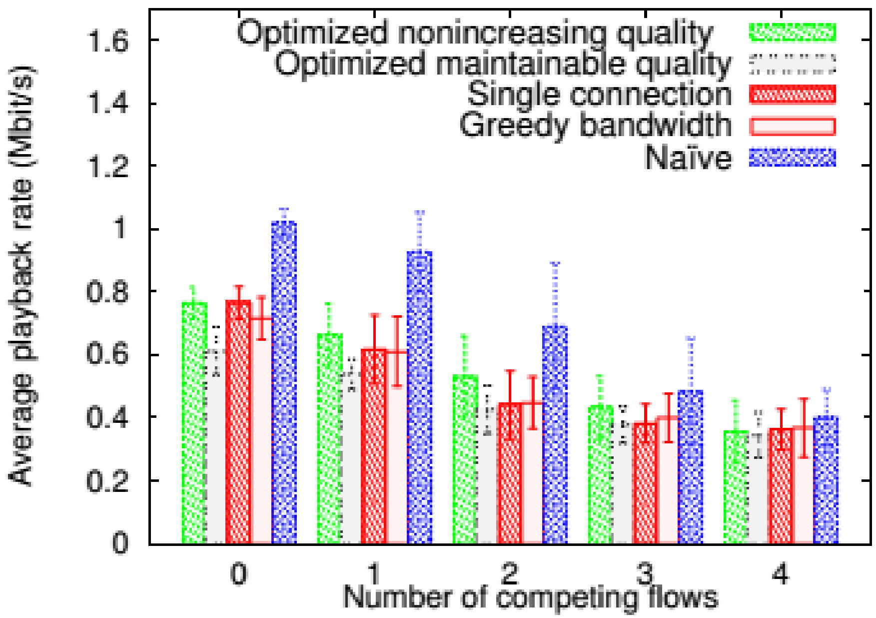


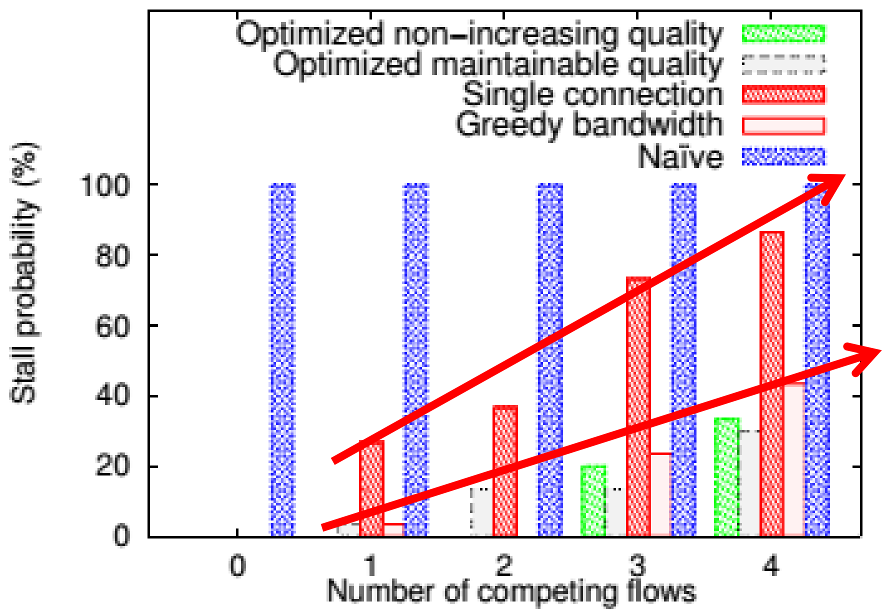


- Player adapts playback rate based on competing traffic
- Parallel connection polices see increased benefits when competing traffic
  - E.g., Single connection policy has much more stalls when competing flows

Impact of Competing Flows

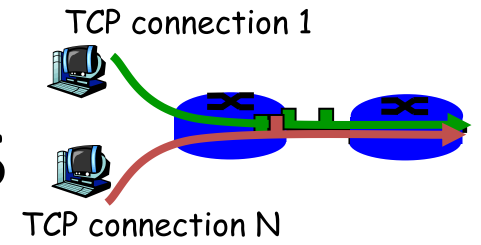


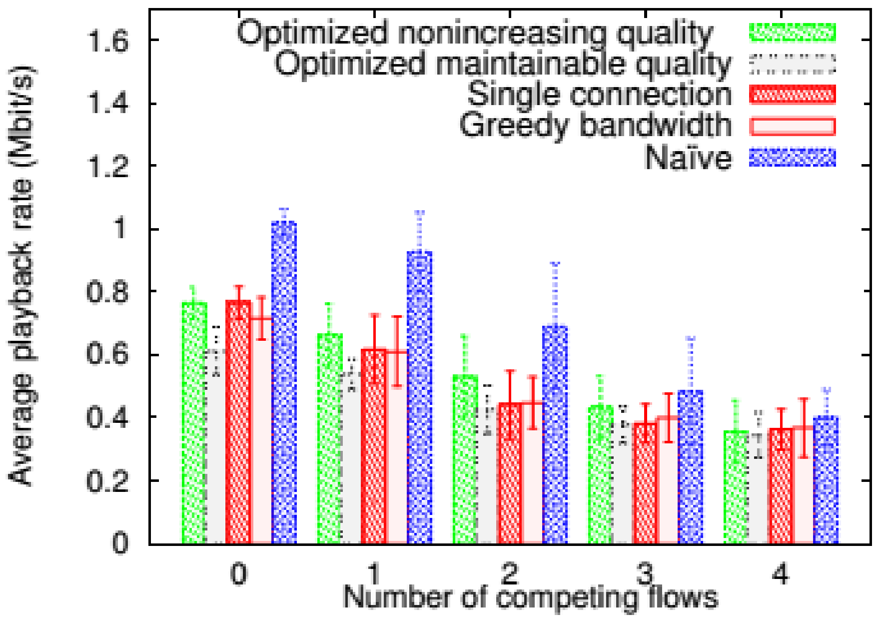


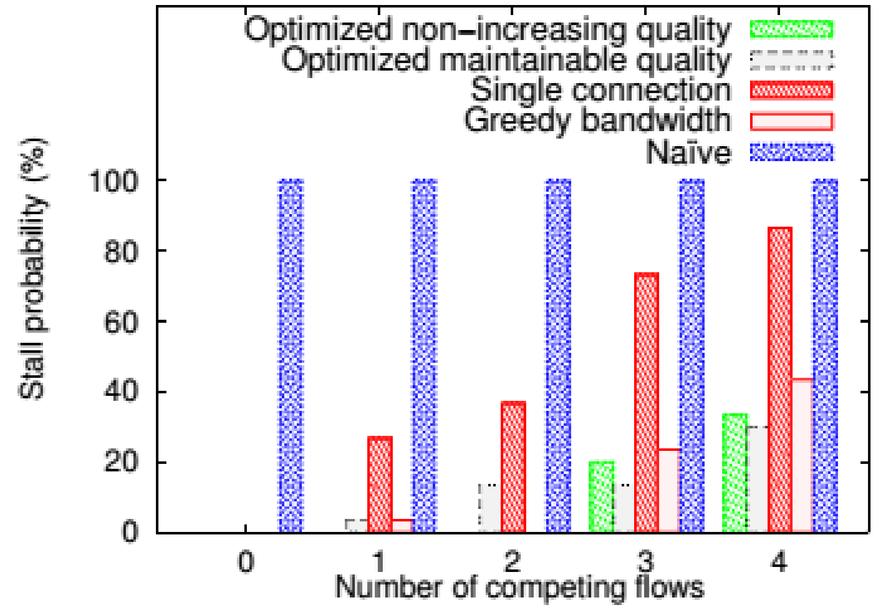


- Player adapts playback rate based on competing traffic
- Parallel connection polices see increased benefits when competing traffic
  - E.g., Single connection policy has much more stalls when competing flows

Impact of Competing Flows



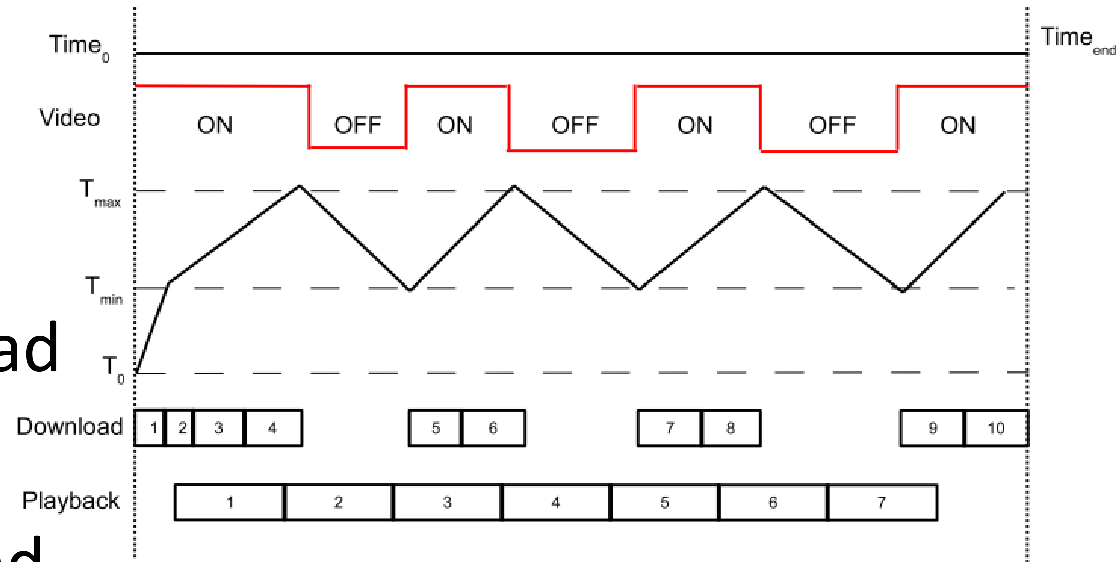




- Player adapts playback rate based on competing traffic
- Parallel connection polices see increased benefits when competing traffic
  - E.g., Single connection policy has much more stalls when competing flows

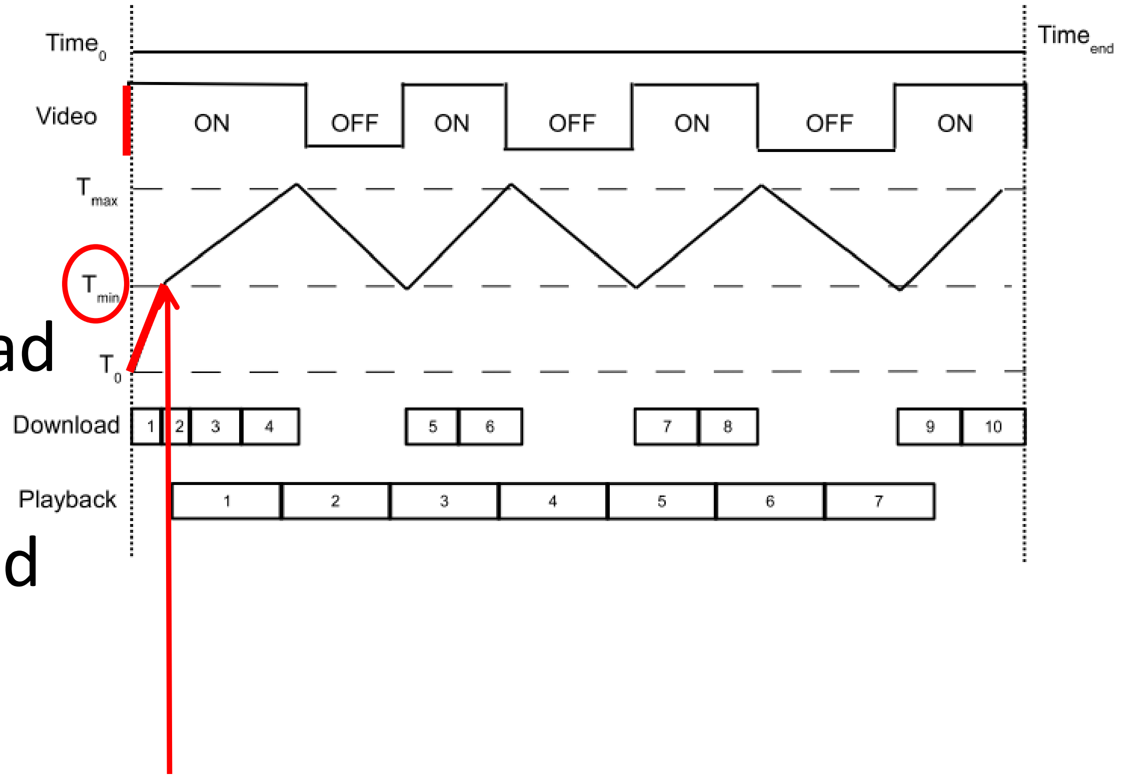
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - – Start playback
- If buffer > T_{max}
  - – Suspend download
- If buffer < T_{min}
  - – Resume download



Capped Workahead

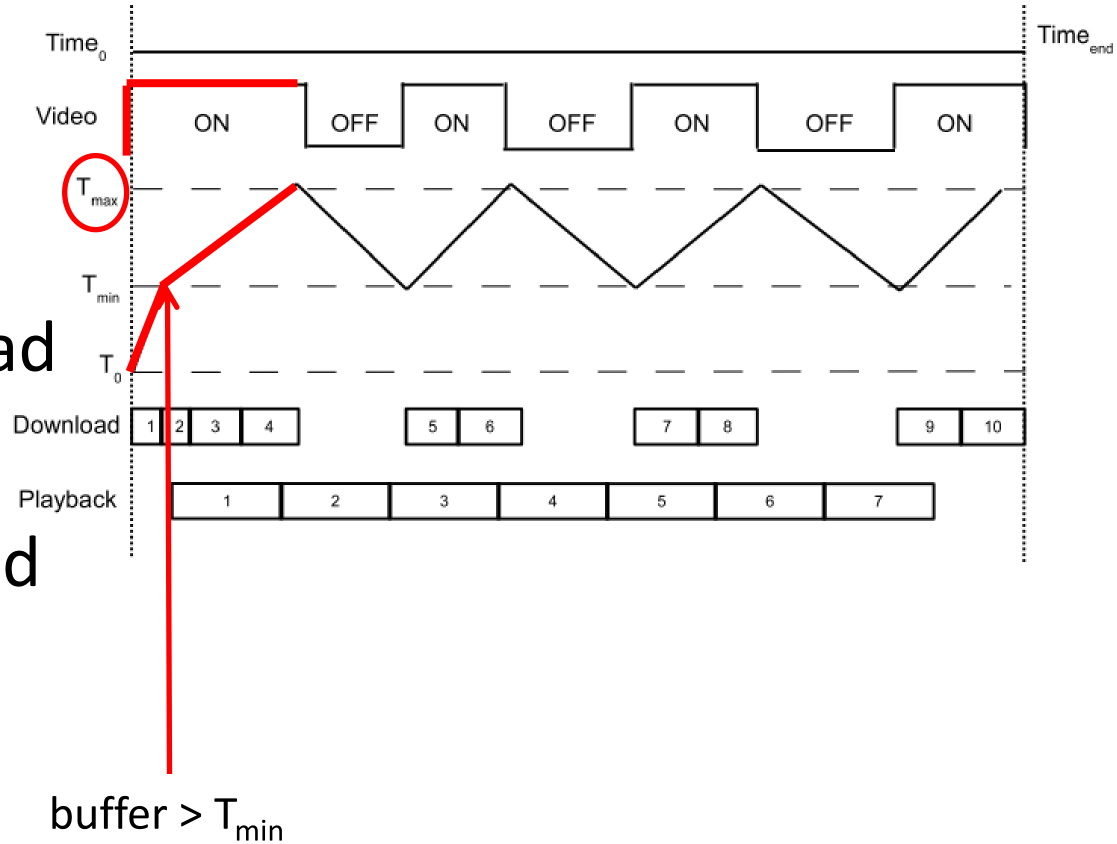
- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



buffer > T_{min}

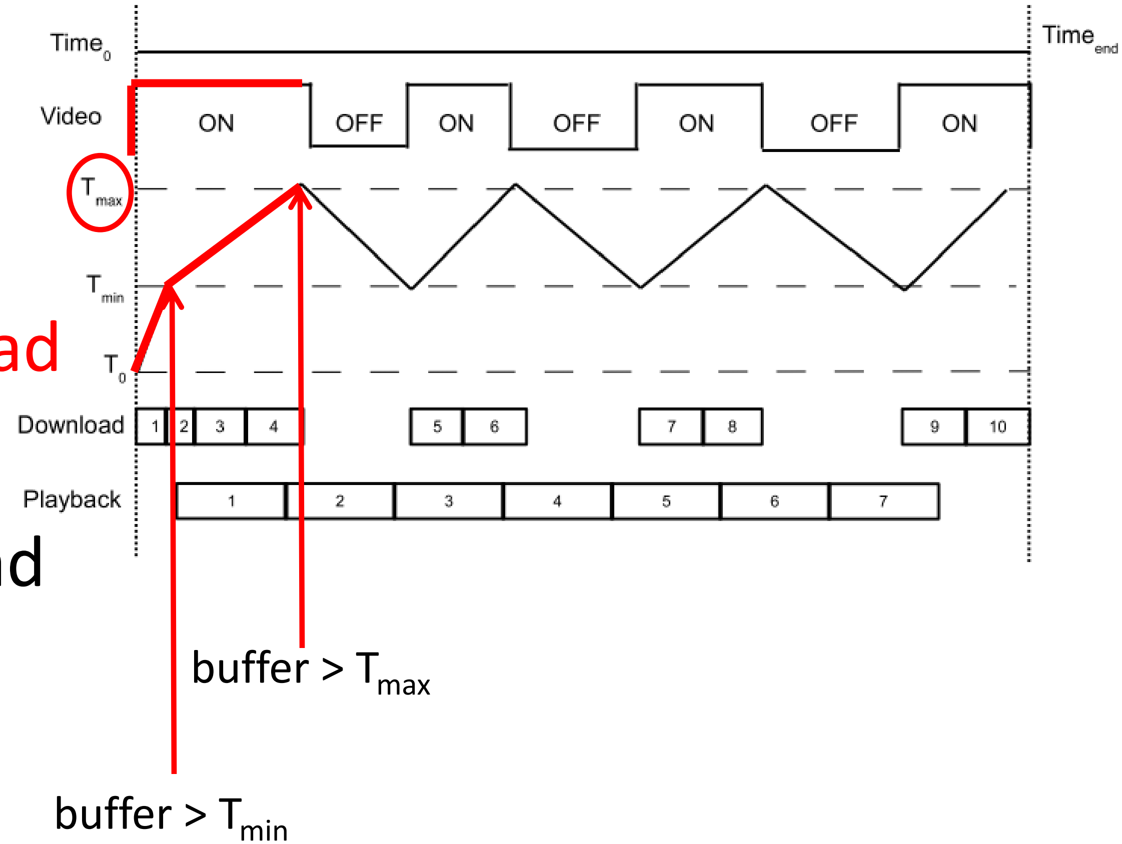
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  – Start playback
- If buffer > T_{max}
  – Suspend download
- If buffer < T_{min}
  – Resume download



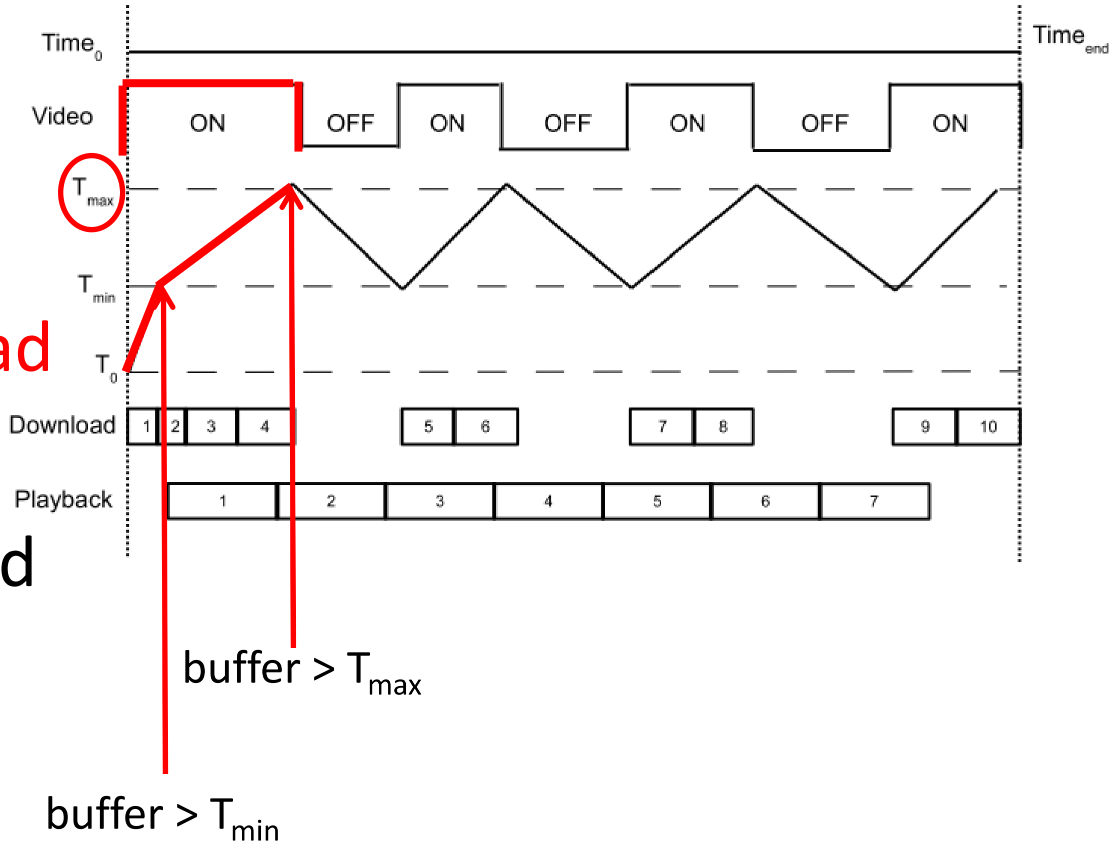
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



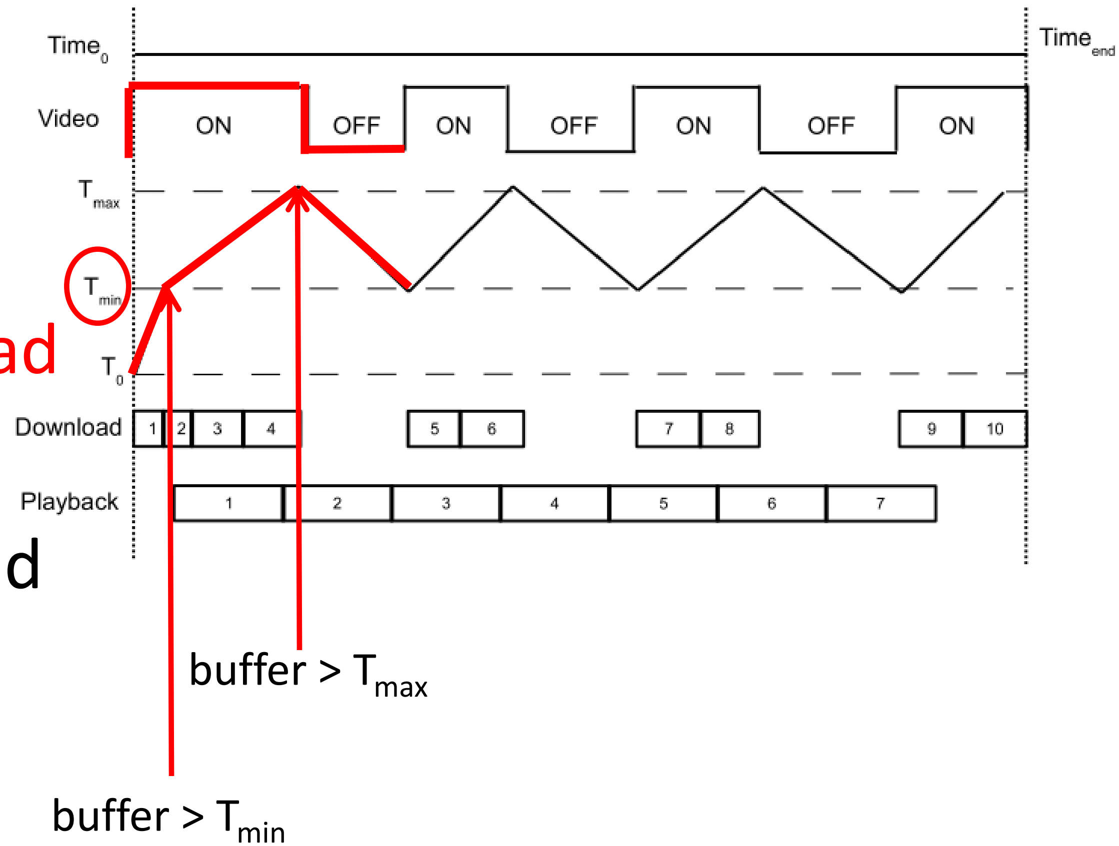
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



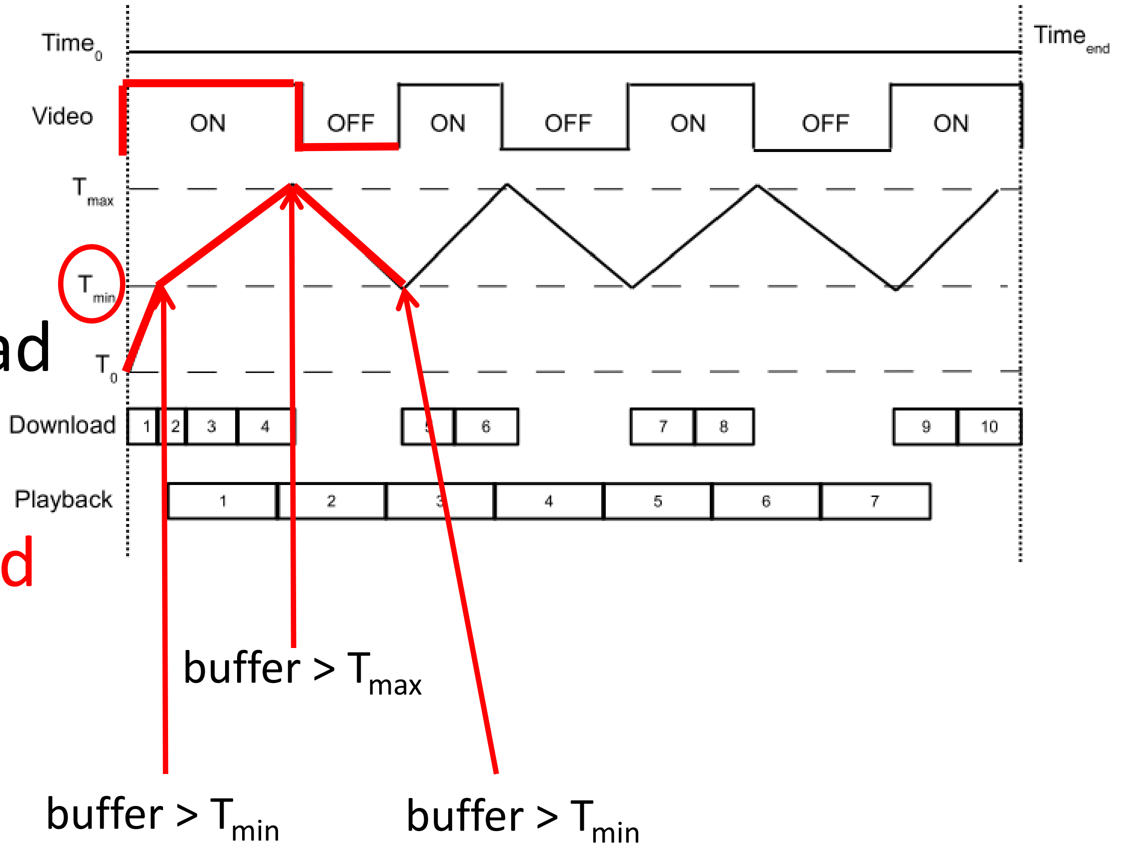
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



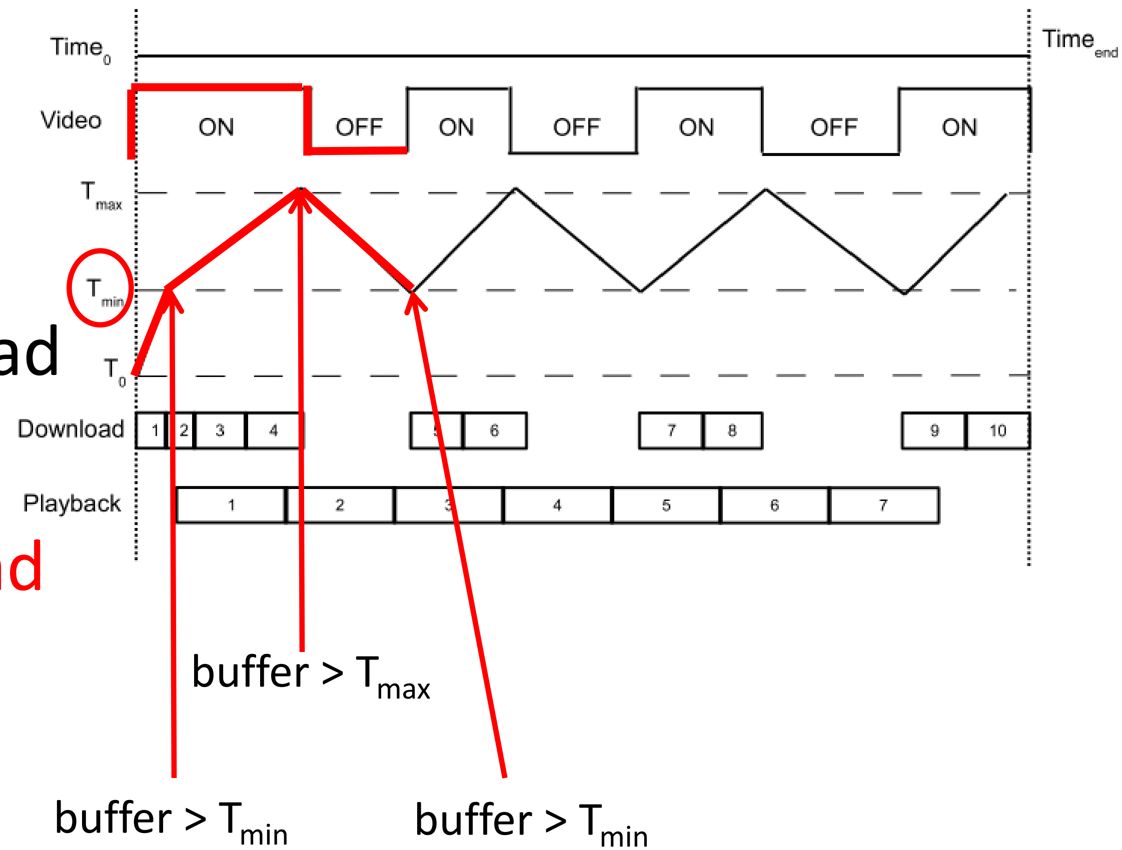
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



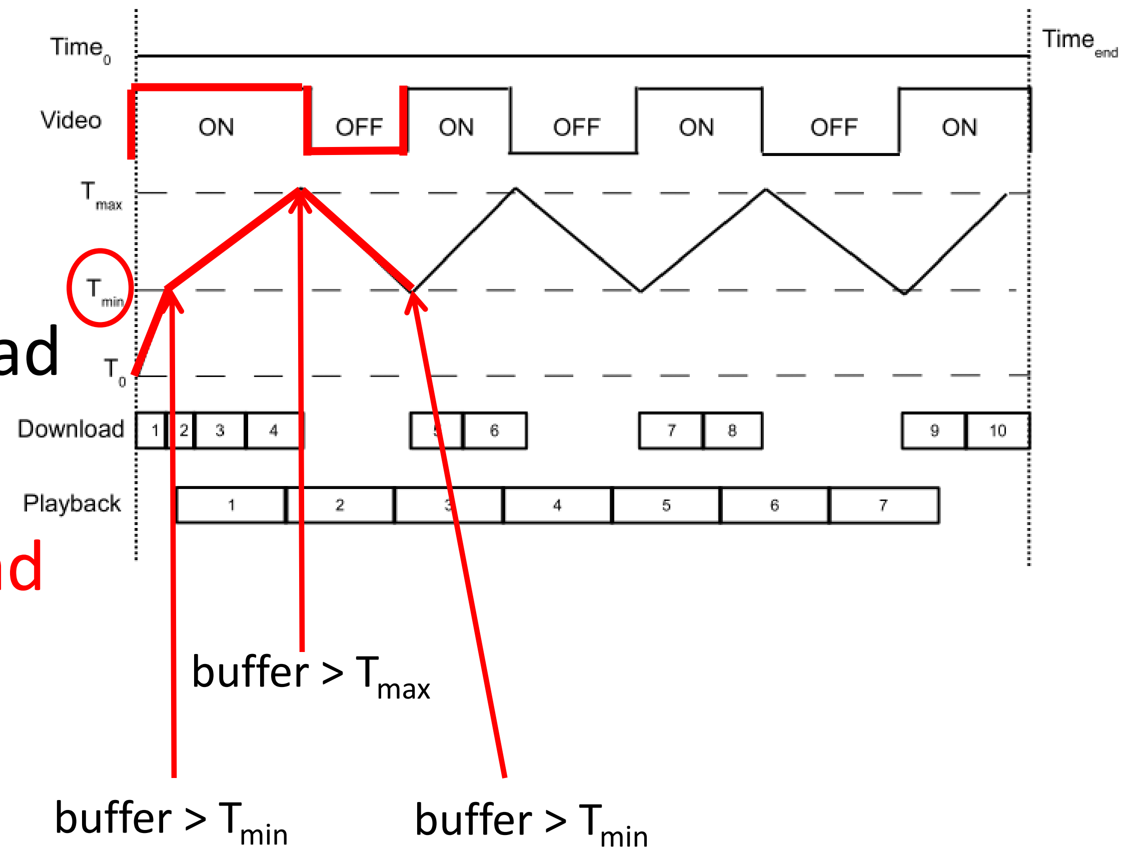
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



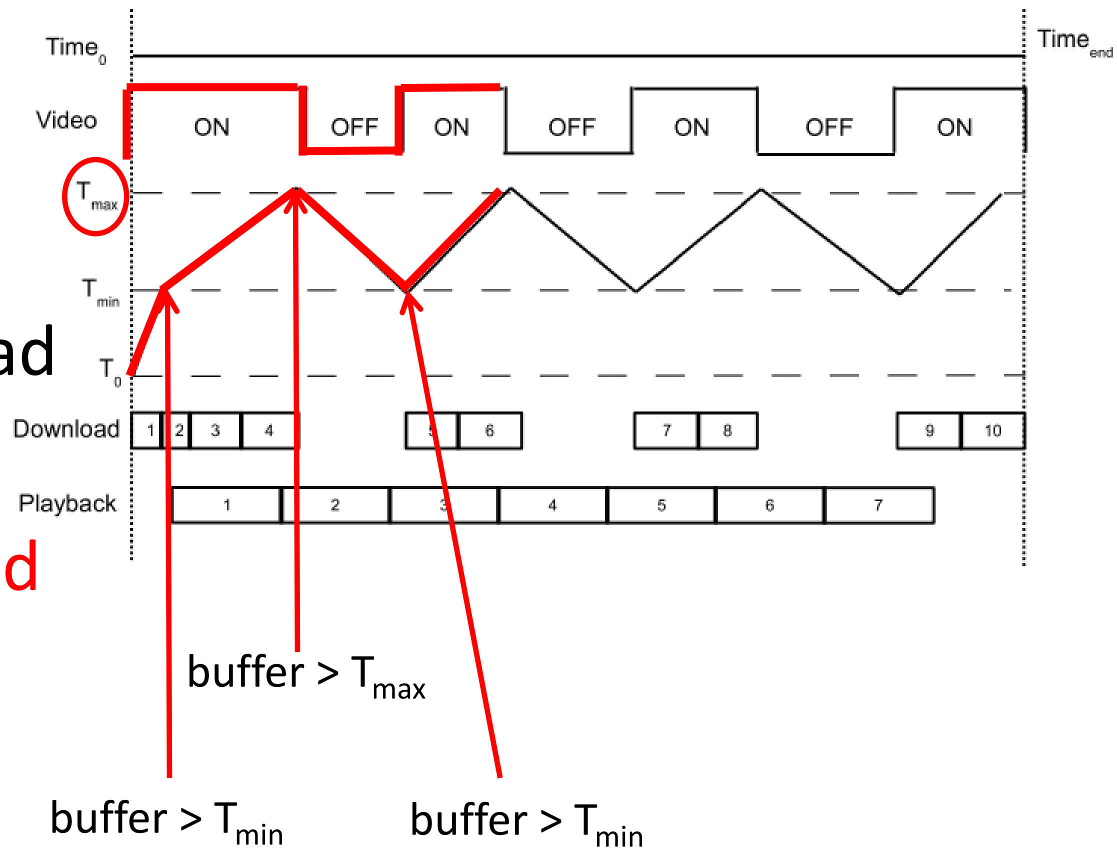
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



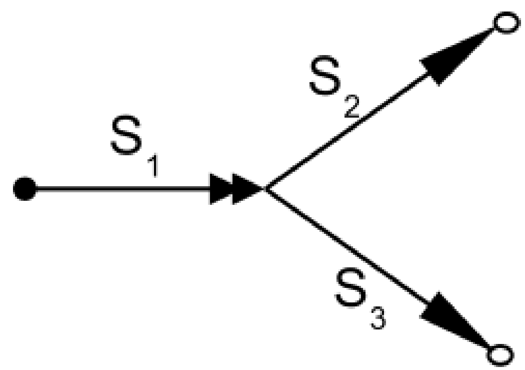
Capped Workahead

- Most HAS players perform ON-OFF switching based on two buffer thresholds: T_{min} and T_{max}
- If buffer > T_{min}
  - Start playback
- If buffer > T_{max}
  - Suspend download
- If buffer < T_{min}
  - Resume download



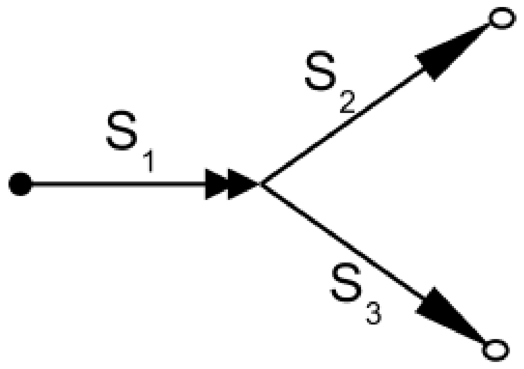
Capped Workahead

- How to handle workahead when video contains branches?



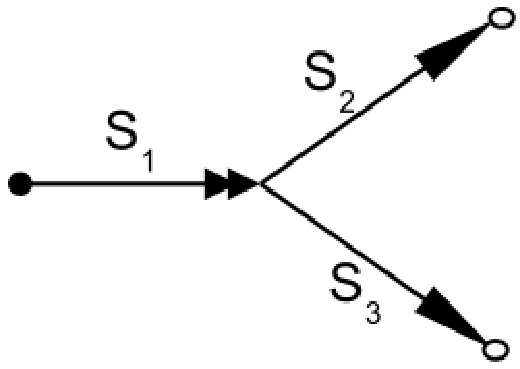
Capped Workahead

- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
  - $T_{min} = T_{single} \cdot$ (# branches)
  - $T_{max} = T_{single} + \Delta$



Capped Workahead

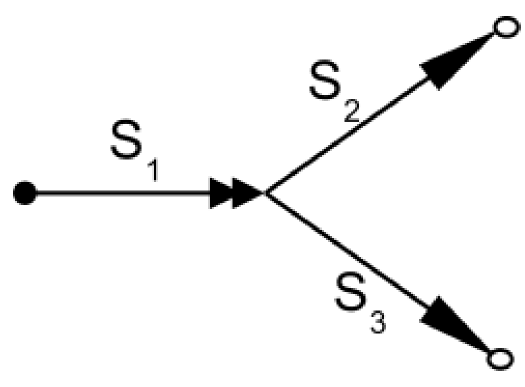
- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
  - $T_{min} = T_{single} \cdot$ (# branches)
  - $T_{max} = T_{single} + \Delta$



Capped Workahead

- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
    - $T_{min} = T_{single} \cdot$ (# branches)
    - $T_{max} = T_{single} + \Delta$

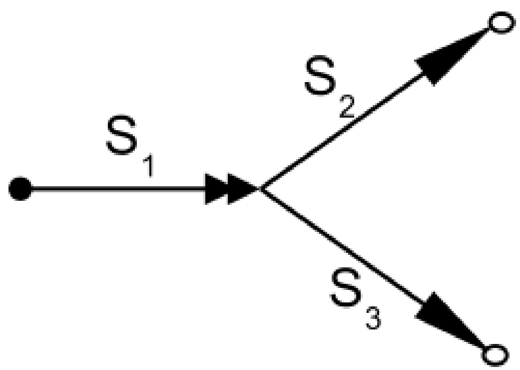
Example



Capped Workahead

- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
  - $T_{min} = T_{single} \cdot$ (# branches)
  - $T_{max} = T_{single} + \Delta$

Example

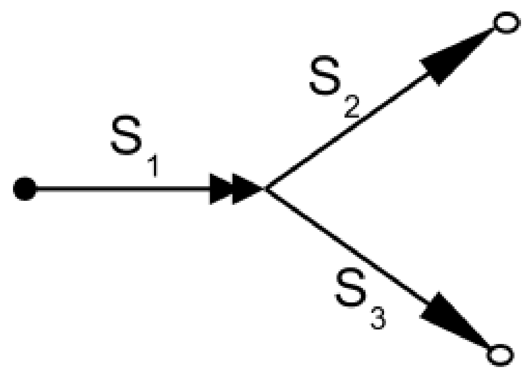


$T_{single} = 8$
$\Delta = 4$
branches = 2

Capped Workahead

- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
  - $T_{min} = T_{single} \cdot$ (# branches)
  - $T_{max} = T_{single} + \Delta$

Example



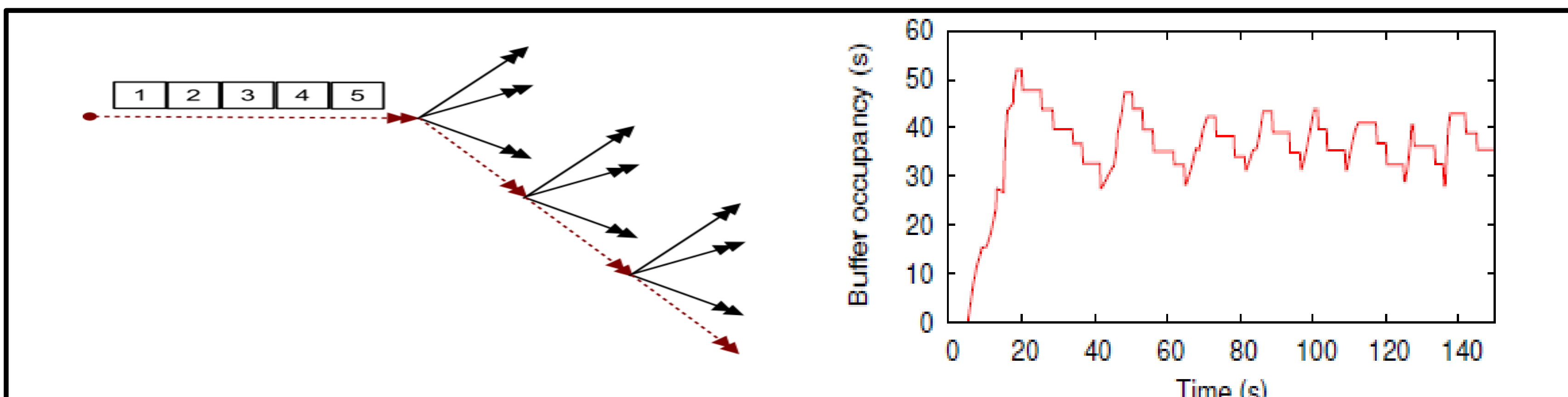
$T_{single} = 8$
$\Delta = 4$
branches = 2

$T_{min} = 16$
$T_{max} = 20$

Capped Workahead

- How to handle workahead when video contains branches?
- Perform ON-OFF switching based on number of branches after the closest branch point
  - $T_{min} = T_{single} \cdot$ (# branches)
  - $T_{max} = T_{single} + \Delta$

Example



Conclusion

- Designed and implemented branched video player that achieve seamless streaming without playback interruptions
- Designed optimized policies that maximize playback quality while ensuring sufficient workahead to avoid stalls
- Evaluation shows that solution effectively adapt quality levels and number of parallel connections so as to provide best possible video quality, given current conditions

Conclusion

- Designed and implemented branched video player that achieve seamless streaming without playback interruptions
- Designed optimized policies that maximize playback quality while ensuring sufficient workahead to avoid stalls
- Evaluation shows that solution effectively adapt quality levels and number of parallel connections so as to provide best possible video quality, given current conditions

Conclusion

- Designed and implemented branched video player that achieve seamless streaming without playback interruptions
- Designed optimized policies that maximize playback quality while ensuring sufficient workahead to avoid stalls
- Evaluation shows that solution effectively adapt quality levels and number of parallel connections so as to provide best possible video quality, given current conditions

Conclusion

- Designed and implemented branched video player that achieve seamless streaming without playback interruptions
- Designed optimized policies that maximize playback quality while ensuring sufficient workahead to avoid stalls
- Evaluation shows that solution effectively adapt quality levels and number of parallel connections so as to provide best possible video quality, given current conditions

Conclusion

- Designed and implemented branched video player that achieve seamless streaming without playback interruptions
- Designed optimized policies that maximize playback quality while ensuring sufficient workahead to avoid stalls
- Evaluation shows that solution effectively adapt quality levels and number of parallel connections so as to provide best possible video quality, given current conditions

Software: http://www.ida.liu.se/~nikca/mm14.html

Quality-adaptive Prefetching for Interactive Branched Video using HTTP-based Adaptive Streaming

Vengatanathan Krishnamoorthi, Niklas Carlsson, Derek Eager,
Anirban Mahanti, Nahid Shahmehri
Linköping University
NICTA
University of Saskatchewan

Software: http://www.ida.liu.se/~nikca/mm14.html

www.liu.se
LiU EXPANDING REALITY